

Dr. Dobb's Journal of Software Tools

FOR THE PROFESSIONAL PROGRAMMER

OBJECT-ORIENTED DESIGN

TOOLS FOR HANDLING

- Binary Trees
- Huge Arrays
- EGA Fonts



REVIEWS

CodeView

Turbo Pascal 4.0



Paradox is the best

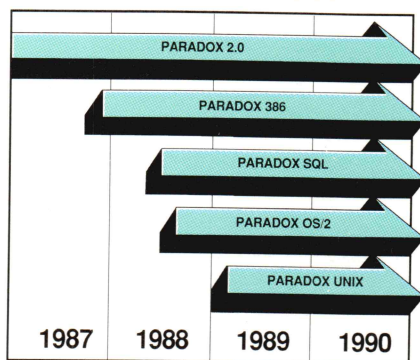
There's no power like Paradox Power

PROGRAM	Overall Power	Overall Versatility	Single-Record Search (sec.)	Indexed Group Record Search (sec.)	Search without an Index (sec.)	Subtotal on Two-File Join (sec.)	Subtotal on Three-File Join (sec.)	Three-File Join (sec.)	Four-File Join (sec.)	Many-to-Many Join (sec.)	Paradox		dBASE III PLUS										
											8.4	9.2	7.5	1.0	6	100	221	325	459	364	322	18	7.9

Source: Software Digest*

Paradox saves you from future shock

Trends for the future with Paradox



Paradox 386 allows users to take advantage of 16 Megabytes of Memory on a 386 machine. This allows Paradox users to work with databases that could in the past only be handled by minicomputers and mainframes.

Your investment today in Paradox applications is protected as new hardware and operating systems are used in your company. Paradox 2.0 applications will run unchanged on Paradox 386, Paradox OS/2, Paradox Unix and Paradox SQL! All versions of Paradox will be completely application and menu compatible. Paradox SQL will allow access to remote databases via SQL. Users will just type in a query as they normally would, and Paradox will translate that Query to SQL.

“ Paradox 2.0 will do for the LAN what the spreadsheet did for the PC

David Schulman,
Bendix Aerospace ”

Paradox makes your network run like clockwork

Paradox is just as valuable to multi and network users as it is to single users. It runs smoothly, intelligently and so transparently that multiusers can access the same data at the same time—without either being aware of each other or getting in each other's way. It works exactly the same way whether you're flying solo or as part of the crew.

“ Anyone who hasn't seen the network version of Paradox should take a look. Ansa has dramatically advanced the state of the art in multiuser network databases

Phil Lemmons,
BYTE

Paradox was a delight to use, both as a stand-alone product and from a local area network server.

Don Crabb,
InfoWorld ”

How to make your network network

To run Paradox 2.0 or the Paradox Network Pack on a network you need:

- Novell with Novell Advanced Netware version 2.0A or higher
- 3Com 3Plus with 3Com 3+ operating system version 1.0, 1.1 or higher
- IBM Token Ring or PC Network with IBM PC Local Area Network Program version 1.12 or higher
- Tonus Tapestry version 1.4 or higher
- AT&T Starlan Network with AT&T PC 6300 Network Program version
- Other network configurations that are 100% compatible with DOS 3.1 and one of the listed networks

System Requirements for Single User:

- DOS 2.0 or higher
- IBM® PS/2 and PC, Compaq® PC families and other 100% compatibles
- 512K RAM
- Two disk drives, 3½-inch and 5¼-inch supported
- Compatible monochrome, color, or EGA monitor with adapter

System Requirements for the Network Workstation:

- DOS 3.1 or higher
- 640K RAM
- Any combination of hard, floppy, or no disk drives
- Compatible monochrome, color, or EGA monitor with adapter

Optional Equipment:

- EMS and EEMS Boards: AST Rampage Board,™ Intel Above Board* or other expanded memory adapters
- Printers: Compatible dot matrix, letter quality, or laser printer

*Reprinted with permission by Software Digest from its July 1987 report covering 12 relational database programs.

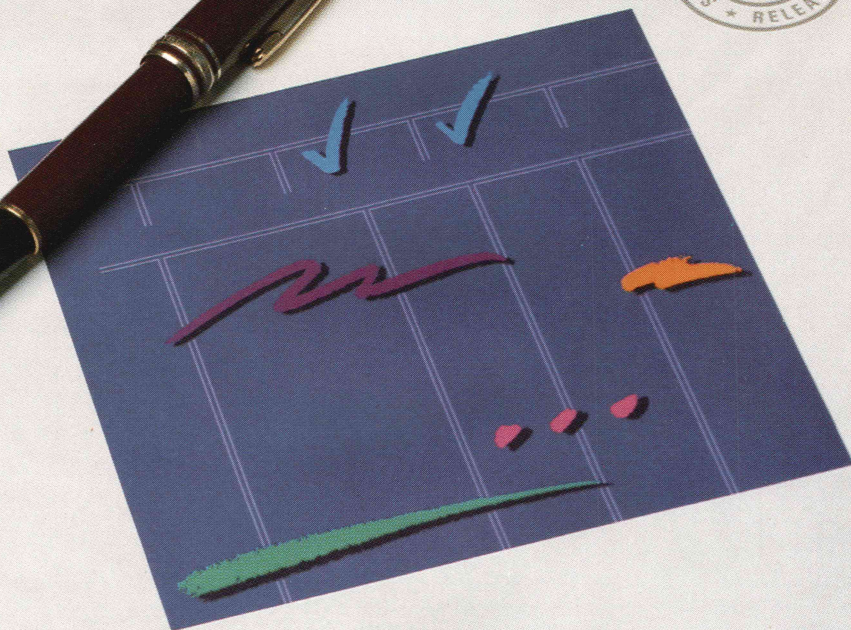
†Test was designed and executed by NSTL. A 1,000-record and a 10,000-record file were joined. A short text field from the 1,000-record file and a numeric field from the 10,000-record file were selected (using the 1,000-record file indexes). The short text field was grouped and sorted in ascending order, the numeric field was subtotaled for each group, and the results output to a null printer. Test times from the last keystroke on the command sequence until return of program control were recorded and averaged.

Paradox is a registered trademark of Ansa Software. Ansa is a Borland International company. Other brand and product names are trademarks or registered trademarks of their respective holders. Copyright ©1988 Borland International, Inc. BI 1205

Paradox:
the top-rated
relational
database
manager
in the world...

PARADOX®

Introduction



Ansa

Why Paradox

"Paradox® is once again the top-rated program, with the latest version scoring even higher than last year's top score." (Software Digest's July 1987 Ratings Report—an independent comparative ratings report for selecting IBM PC Business software).

All tests for the Ratings Report were done by the prestigious National Software Testing Laboratory, Philadelphia, PA, and the message is crystal clear: there is no better relational database manager than Paradox.

NSTL tested 12 different programs and amongst other results, discovered that Paradox is 3 times faster than dBASE; 6 times faster than R:BASE on a two-file join with subtotals test†.

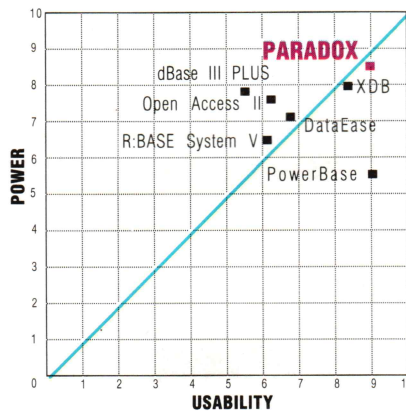
Paradox does the impossible: combines ease-of-use with power and sophistication

Even if you're a beginner, Paradox is the only relational database manager that you

can take out of the box and begin using right away.

Because Paradox employs state-of-the-art artificial intelligence technology, it does almost everything for you—except take itself out of the box.

If you've ever used 1-2-3® or dBASE®, you already know how to use Paradox. It has Lotus-like menus, and Paradox documentation includes "A Quick Guide to Paradox for Lotus users," and "A Quick Guide to Paradox for dBASE users."



Source: Software Digest*

Ideal programs have high levels of both power and usability. Programs plotted in the upper righthand portion of the diagram above come closest to achieving that ideal.

“Paradox still offers superior import/export facilities using Lotus 1-2-3, dBASE, ASCII and other file types. It transfers between formats with stunning speed

Rusel DeMaria, PC Week ”

Paradox responds instantly to "Query-by-Example"

The method you use to ask questions is called Query-by-Example. Instead of spending time figuring out *how* to do the query, you simply give Paradox an example of the results you're looking for. Paradox picks up the example and automatically seeks the fastest way of getting the answer. Paradox, unlike other databases, makes it just as easy to query multiple tables simultaneously as it is to query one.

Source: Software Digest*		Software Digest Rating	Program Name	Version Tested		Ease of Learning		Ease of Use		Error Handling	Performance	Versatility	Memory Requirement	Price
Overall Evaluation														
☆☆☆☆	8.7	Paradox	1.1	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	512K	\$495		
☆☆☆☆	8.2	XDB	1.10	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	320K	\$750		
☆☆☆	7.6	PowerBase	2.3	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	384K	\$349		
☆☆☆	7.0	Open Access II	2.0	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	256K	\$395		
☆☆☆	7.0	DataEase	2.5/2	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	384K	\$600		
☆☆	6.6	dBASE III PLUS	1.1	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	384K	\$695		
☆☆	6.4	R:BASE System V	1.1	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	512K	\$700		

RATINGS KEY

(On a scale of 0 to 10)

Overall Evaluation

☆☆☆☆ 9.0 or higher

☆☆☆☆ 8.0 - 8.9

☆☆☆ 7.0 - 7.9

☆☆ 6.0 - 6.9

☆ 5.0 - 5.9

All Other Ratings

7.0 - 9.9

5.0 - 6.9

UNDER 5.0

RATINGS KEY
(On a scale of 0 to 10)
Overall Evaluation
☆☆☆☆ 9.0 or higher
☆☆☆☆ 8.0 - 8.9
☆☆☆☆ 7.0 - 7.9
☆☆ 6.0 - 6.9
☆☆ 5.0 - 5.9

All Other Ratings
7.0 - 9.9
5.0 - 6.9
UNDER 5.0

Paradox: the new corporate standard

Paradox automatically updates your data and lets you control access to information

In "Co-Edit" mode, changes made by anyone are automatically updated to everyone. You can pre-set a "Screen-Refresh" interval to occur anywhere from 1-second to 1-hour intervals. (If you don't make a pre-set choice, Paradox automatically updates every 3 seconds so that your screen always shows you updated data).

While Paradox 2.0 lets everyone share and update information simultaneously, you can configure it to keep secrets secret.

You can restrict others' rights in a variety of ways with safeguards protecting confidential files and/or giving someone "Read Only" rights which is to allow "View," but prevent "Change." The Paradox technique—automatic file and record locking—ensures data accuracy and integrity in any multiuser environment.

For a brochure or the dealer nearest you call (800) 543-7543

“ With Version 2.0, Paradox becomes a sophisticated multiuser product that boasts an impressive selection of data-protection features and password-security levels

Rusel DeMaria,
PC Week ”

Get serious support for serious Paradox application programming

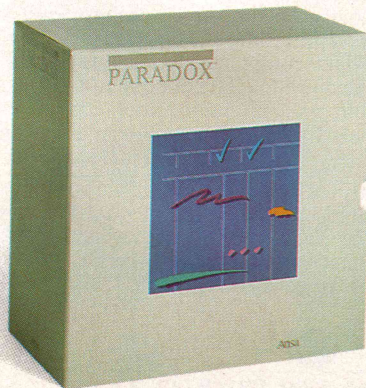
When you subscribe to the Paradox Developer's Resource Program (PDRP), you get all the resources and support you need for sophisticated Paradox application development: unlimited access to our toll-free PAL programmers support line; the Paradox Developer's Toolkit; a subscription to *Turbo Technix*, Borland's bi-monthly technical magazine; and a 20% discount on the Paradox User's Journal published by the Cobb Group.

Call our Customer Service Department at (408) 438-8400 for your free PDRP information kit with all the details.

PARADOX

by Ansa

A Borland Company



Interlocking Pieces: Blaise and Turbo Pascal.

Now, for
Turbo Pascal 4.0!

THE BLAISE M E N U

Whether you're a Turbo Pascal expert or a novice, you can benefit from using professional tools to enhance your programs. With Turbo POWER TOOLS PLUS™ and Turbo ASYNCH PLUS™, Blaise Computing offers you all the right pieces to solve your 4.0 development puzzle.

Compiled units (TPU files) are provided so each package is ready to use with Turbo Pascal 4.0. Both POWER TOOLS PLUS and ASYNCH PLUS use units in a clear, consistent and effective way. If you are familiar with units, you will appreciate the organization. If you are just getting started, you will find the approach an illustration of how to construct and use units.

◆ **POWER TOOLS PLUS** is a library of over 180 powerful functions and procedures like fast direct video access, general screen handling including multiple monitors, VGA and EGA 50-line and 43-line text mode, and full keyboard support, including the 101/102-key keyboard. Stackable and removable windows with optional borders, titles and cursor memory provide complete windowing capabilities. Horizontal, vertical, grid and Lotus-style menus can be easily incorporated into your programs using the menu management routines. You can create the same kind of moving pull down menus that Turbo Pascal 4.0 uses.

Control DOS memory allocation. Alter the Turbo Pascal heap size when your program executes. Execute any program from within your program and POWER TOOLS PLUS automatically compresses your heap memory if necessary. You can even force the output of the program into a window!

Write general interrupt service routines for either hardware or software interrupts. Blaise Computing's unique intervention code lets you develop memory resident (TSRs) applications that take full advantage of DOS capabilities. With simple procedure calls, "schedule" a Turbo Pascal procedure to execute either when pressing a "hot key" or at a specified time.

◆ **ASYNCH PLUS** provides the crucial core of hardware interrupts needed to support asynchronous data communications. This package offers simultaneous buffered input and output to both COM ports, and up to four ports on PS/2 systems. Speeds to 19.2K baud, XON/XOFF protocol, hardware handshaking, XMODEM (with CRC) file transfer and modem control are all supported. ASYNCH PLUS provides text file device drivers so you can use standard "Readln" and "Writeln" calls and still exploit interrupt-driven communication.

The underlying functions of ASYNCH PLUS are carefully crafted in assembler and drive the hardware directly. Link these functions directly to your application or install them as memory resident.

Blaise Computing products include all source code that is efficiently crafted, readable and easy to modify. Accompanying each package is an indexed manual describing each procedure and function in detail with example code fragments. Many complete examples and useful utilities are included on the diskettes. The documentation, examples and source code reflect the attention to detail and commitment to technical support that have distinguished Blaise Computing over the years.

Designed explicitly for Turbo Pascal 4.0, Turbo POWER TOOLS PLUS and Turbo ASYNCH PLUS provide reliable, fast, professional routines—the right combination of pieces to put your Turbo Pascal puzzle together. **Complete price is \$129.00 each.**

Turbo POWER SCREEN \$129.00
NEW! General screen management; paint screens; block mode data entry or field-by-field control with instant screen access. Now for Turbo Pascal 4.0, soon for C and BASIC.

Turbo C TOOLS \$129.00
Full spectrum of general service utility functions including: windows; menus; memory resident applications; interrupt service routines; intervention code; and direct video access for fast screen handling. For Turbo C.

C TOOLS PLUS \$129.00
Windows; menus; ISRs; intervention code; screen handling and EGA 43-line text mode support; direct screen access; DOS file handling and more. Specifically designed for Microsoft C 5.0 and QuickC.

ASYNCH MANAGER \$175.00
Full featured interrupt driven support for the COM ports. I/O buffers up to 64K; XON/XOFF; up to 9600 baud; modem control and XMODEM file transfer. For Microsoft C and Turbo C or MS Pascal.

PASCAL TOOLS/TOOLS 2 \$175.00
Expanded string and screen handling; graphics routines; memory management; general program control; DOS file support and more. For MS-Pascal.

KeyPilot \$49.95
"Super-batch" program. Create batch files which can invoke programs and provide input to them; run any program unattended; create demonstration programs; analyze keyboard usage.

EXEC \$95.00
NEW VERSION! Program chaining executive. Chain one program from another in different languages; specify common data areas; less than 2K of overhead.

RUNOFF \$49.95
Text formatter for all programmers. Written in Turbo Pascal; flexible printer control; user-defined variables; index generation; and a general macro facility.

TO ORDER CALL TOLL FREE
800-333-8087

TELEX NUMBER - 338139

BLAISE COMPUTING INC.

2560 Ninth Street, Suite 316 Berkeley, CA 94710 (415) 540-5441

CIRCLE NO. 102 ON READER SERVICE CARD

YES! Send me the right pieces!
Enclosed is \$_____ for _____ copies of _____
☐ Please send me more information on your products.
CA residents add Sales Tax. Domestic orders add \$4.00 for UPS shipping, \$10.00 for Federal Express standard air.
Name: _____ Phone: (____) _____
Address: _____ State: _____ Zip: _____
City: _____ Exp. Date: _____
VISA or MC#: _____

Microsoft and QuickC are registered trademarks of Microsoft Corporation. Turbo Pascal is a registered trademark of Borland International.

ARTICLES

Object-Oriented ►**Object-Oriented Programming and Database Design 18***by Jacob Stein*

Object-oriented programming has been gaining acceptance in the last few years. In this article, Jacob gives an introduction to object-oriented programming in general, a description of a specific object-oriented language (OPAL), and some example problems that are particularly well suited to object-oriented solutions.

EGA ►**Writing Custom Display Fonts 36***by Andrew J. Chalk*

Andrew describes a TSR he's written for the EGA that automatically replaces the standard screen font with a resident custom font.

C routines ►**Threaded Binary Trees 42***by James Mathews*

Textbook discussions of threaded binary trees are almost always accompanied by source code in Pascal, aimed at the university student. In this article, Jim describes some routines he developed for traversing and managing these trees in Microsoft C.

REVIEWS

EXAMINING ROOM 116*coordinated by Ron Copeland*

Products examined from the programmer's perspective. This month's offering includes a detailed review of Borland's Turbo Pascal 4.0, and a quick look at Microsoft's CodeView debugger.

COLUMNS

TO THE MACS 90*by Stan Krute*

Stan winds up the CDEF project he began in the January column, and talks about several products.

Pascal ►**STRUCTURED PROGRAMMING 108***by Kent Porter*

Not all problems are polite enough, or trivial enough, to shoehorn into the PC's 64K segments. This month Kent shares his techniques for handling large data arrays in Turbo Pascal.

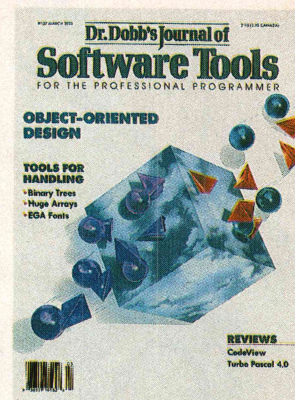
FORUM

EDITORIAL 6*by Tyler Sperry***RUNNING LIGHT 8***by Tyler Sperry***LETTERS 12***by you***CARTOON 14***by Joe Sikoryak***SWAINE'S FLAMES 136***by Michael Swaine*PROGRAMMER'S
SERVICES**ADVERTISER INDEX: 81**

Where to go for more information on products.

OF INTEREST 128

Product news for the programming community.

**About the Cover**

If nothing else, object-oriented languages have gotten a rap for being abstract and not as efficient as needed for real-world problems. This month's lead article discusses object-oriented programming languages and some real-world problems where the object-oriented approach excels. Sometimes all you need are the right tools (like data hiding and inheritance) and a little "blue sky" thinking.

Next Issue

The annual AI issue, with language hacking techniques for fanciers of LISP and PROLOG, a review of Objective-C, and all the usual foolishness.

NEW DATABASE
OS/2 API
AVAILABLE NOW!

Finally. A pro for people who h

Nobody ever said programming PCs was supposed to be easy.

But does it have to be tedious and time-consuming, too?

Not any more.

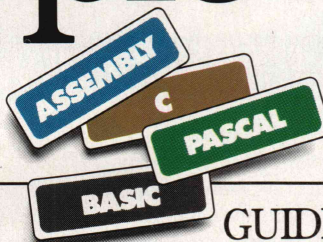
Not since the arrival of the remarkable new program in the lower right-hand corner.

Which is designed to save you most of the time you're currently spending searching through the books and manuals on the shelf above.

The Norton On-Line Programmer's Guides are a quartet of pop-up reference packages that do the same things in four different languages.

Each package consists of two parts: A memory-resident instant access program. And a comprehensive, cross-referenced database crammed with just about everything you need to

know to program in your favorite language.



GUIDES DATA

Instant Access Program

- Memory-resident—uses just 71K.
- Full-screen or moveable half-screen view, with pull-down menus.
- Auto lookup and searching.
- Tools for compiling your own databases.

ASSEMBLY (600K of data)

- DOS Service Calls: All INT 21h services, interrupts, error codes, FCB and PSP fields, standard handles and more.
- ROM BIOS Calls: All ROM calls plus low RAM usage.
- Instruction Set: All 8088/86 instructions, addressing modes, flags, bytes per instruction, clock cycles and more.
- MASM: Pseudo-ops and assembler directives.
- Tables: ASCII chart, line-drawing charts, keyboard scan codes and more.

BASIC (270K each database)

- IBM BASIC, Microsoft QuickBASIC and TurboBASIC.
- Statements and Functions: Describes all statements and built-in library functions.

- Tables: Line-drawing characters, ASCII chart, keyboard codes, error codes, operators, etc.

C (600K each database)

- Microsoft C and Turbo C: Describes language, including statements, operators, data types and structures.
- Library Functions: Detailed descriptions of all functions, from abort () to write ().
- Preprocessor Directives: Describes commands, usage and syntax.
- Tables: ASCII chart, line-drawing characters, keyboard codes, error codes, operators, etc.

PASCAL—Turbo (360K of data)

- Language: Describes statements, syntax, operators, data types and records.
- Library: Describes the library procedures and functions.
- Tables: ASCII chart, line-drawing characters, keyboard codes, error codes, reserved words, etc.

(If you don't believe us, you might want to take a moment or two to examine the data box you just passed.)

You can, of course, find most of this



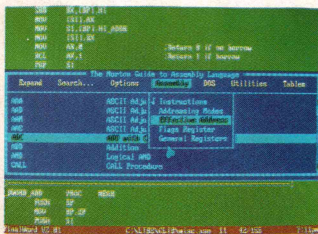
programming tool ate manual labor.

information in the books and manuals on our shelf.

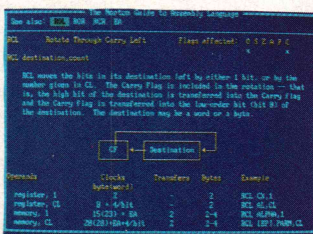
But Peter Norton—who's written a few books himself—figured you'd rather have it on your screen.

In seconds.

In either full-screen or moveable half-



A Guides reference summary screen (shown in blue) pops up on top of the program you're working on (shown in green).



Summary data expands on command into extensive detail. And you can select from a wide variety of information.

screen mode.

Popping up right next to your work. Right where you need it.

This, you're probably thinking, is precisely the kind of thinking that produced the classic Norton Utilities.

And you're right.

But even Peter Norton can't think of

everything.

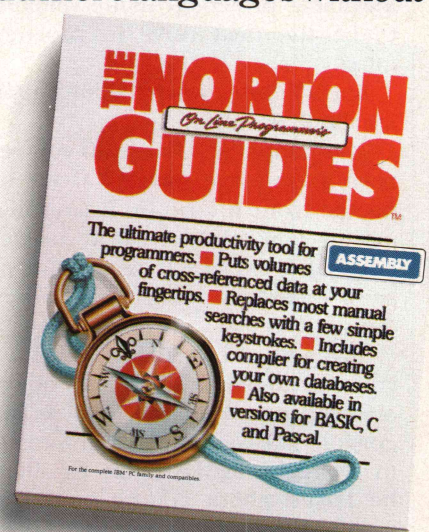
Which is why there's a built-in compiler for creating databases of your own.

And why all Guides databases are compatible with the instant access program in your original package.

So you can add more languages without spending a lot more money.

To get more information, call your dealer. Or call Peter Norton at 1-800-451-0303 Ext. 40.

And ask for some guidance.



Peter Norton
COMPUTING

EDITORIAL

Object Lessons

Recently I attended Microsoft's Annual Systems Software Seminar. This is less a seminar in the academic sense than a carefully orchestrated presentation for the press, a gathering where Microsoftians like Bill Gates and Steve Ballmer take turns at a podium and argue over exactly when OS/2 will take over the known universe. There were also presentations on future enhancements to DOS, the OS/2 LAN and SQL servers, future developer goodies, etc.

Bill Gates remarks on programming languages really got me thinking, though. He spent a fair bit of time explaining to an audience of industry analysts and press people about these things called "objects" and how they were the coming thing in programming. This is a view I quite agreed with. The increasing complexity of environments like the Mac Toolbox and the OS/2 Presentation Manager cry out for new programming tools and methods. Language packages like Actor and Smalltalk/V have the potential (in their inevitable protected mode versions) to be the next wave in programming.

But Bill Gates wasn't talking about Smalltalk or Actor.

The people at Microsoft, you see, have a language division that's expected to turn a profit. It isn't their job to convert the world to new languages, it's to sell language packages. Preferably more than Borland does.

So when asked how Microsoft was going to address the topic of object-oriented programming with its languages, Bill Gates wasn't interested in weirdo languages like Smalltalk; he talked about adding object-oriented extensions to existing Microsoft languages: C and BASIC.

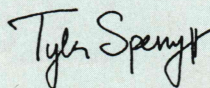
Object-oriented BASIC! What a concept! Soon beginners will be able to add modular meatballs to their spaghetti code. My mind boggled with

the possibilities and almost missed the follow-up from Gates: there's no promise of when these extensions will appear, or in what form. ANSI C is still the chosen language, and converting to C++ seems unlikely.

The next day on the plane home I was still pondering the notion of what—for want of a better name—I'm calling OOBASIC (pronounced "uh-oh BASIC"). The question I was wrestling with is non-trivial: how much of the increased sales of programming languages is based on advances in programming technology, and how much is simply improved marketing? It's a difficult question. After all, Borland had a Modula-2 compiler years ago and dumped it. Yet the same company has managed to make money on PROLOG by hyping it as "the language of artificial intelligence."

I've spent some time considering Bill Gates' opinion that no new languages will make an impact in the years to come, and I'm still not convinced. It seems to me that the new paradigms and new environments will demand new approaches, not patches on old languages. The folks at the Whitewater Group and Digtalk and other companies seem to be able to do quite well marketing solutions rather than trying to jump on the bandwagon and sell yet another ANSI C compiler. Just maybe they're on to something. And maybe object-oriented extensions to older languages are really a step back instead of forward. The next year or so will tell.

In the meantime, I'll be spending my evenings exploring Smalltalk and other languages, examining new paradigms, and counting myself lucky on the days I don't get a press release announcing object-oriented COBOL.



Tyler Sperry
editor

Dr. Dobb's Journal of Software Tools

FOR THE PROFESSIONAL PROGRAMMER

Editorial

Editor-in-Chief	Michael Swaine
Editor	Tyler Sperry
Managing Editor	Monica E. Berg
Associate Editor	Ron Copeland
Assistant Editor	Sara Noah Ruddy
Technical Editors	Allen Holub Richard Relp Kent Porter
Contributing Editors	Stan Krute Martin Tracy
Copy Editor	Rhoda Simmons
Art/Production	
Director	Larry L. Clay
Art Director	Michael Hollister
Assoc. Art Director	Joe Sikoryak
Technical Illustrator	Barbara Mautz
Typesetter	Mary E. Lopez
Cover Artist	Bunny Carter

Circulation

Circulation Director	Maureen Kaminski
Fulfillment Coordinator	Francesca Martin
Subscription Supervisor	Kathleen Shay
Newsstand Coordinator	Sarah Frisbie

Administration

Vice President of Finance and Operations	Kate Wheat
Business Manager	Betty Trickett
Accounts Payable Supv.	Mayda Lopez-Quintana
Accts. Receivable Supv.	Laura DiLazzaro

Marketing/Advertising

Director	Ferris Perdon
Advertising Coordinator	Patricia Albert
Account Managers	see page 81

Publisher

Peter Hutchinson

Dr. Dobb's Journal of Software Tools (USPS 307690) is published monthly by M&T Publishing Inc., 501 Galveston Dr., Redwood City, CA 94063; (415) 366-3600. Second-class postage paid at Redwood City and at additional entry points. DDJ is published under license from People's Computer Company, 2682 Bishop Dr., Suite 107, San Ramon, CA 94583, a nonprofit corporation.

Article Submissions: Send manuscripts and disk (with article and listings) to the Associate Editor.

DDJ on CompuServe: Type GO DDJ

Address Correction Request: Postmaster: Send Form 3579 to Dr. Dobb's Journal, P.O. Box 3713, Escondido, CA 92025.

ISSN 0888-3076

Customer Service: For subscription problems call: outside CA (800) 321-3333; in CA (619) 485-9623 or 566-6947. For book/software order problems call (415) 366-3600.

Subscriptions: \$29.97 per 1 year; \$56.97 for 2 years. Canada and Mexico add \$28 per year airmail or \$11 per year surface. All other countries add \$32 per year airmail. Foreign subscriptions must be prepaid in U.S. funds drawn on a U.S. bank. For foreign subscriptions, TELEX: 752-351.

Foreign Newsstand Distributor: Worldwide Media Service Inc., 386 Park Ave. South, New York, NY 10016; (212) 686-1520 TELEX 620430 (WUI).

Entire contents copyright © 1988 by M&T Publishing, Inc., unless otherwise noted on specific articles. All rights reserved.

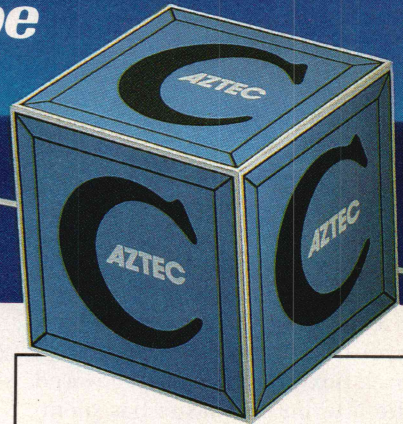


M&T Publishing Inc.

Chairman of the Board	Otmar Weber
Director	C. F. von Quad
President	Laird Foshay
Vice President of Publishing	William P. Howard

Aztec C

*Power to go the distance...
Whatever that distance might be*



From real time embedded applications to comprehensive commercial applications on Macintosh, IBM PC, Amiga, Atari, and others, Aztec C has earned a well-deserved reputation as an innovative, tough to beat, rock-solid C development system.

But don't just take our word for it—try it yourself. We know that the best way to understand what puts you ahead with Aztec C is to use it. That's why Aztec C

systems purchased directly from Manx come with a 30-day, no questions asked, satisfaction guarantee. Call for yours today.

We can also send you information that details the special features and options of Aztec C. Plus information on support software, extended technical support options, and all of the services and specialized support that you may need when you're pushing your software to the limits and ... beyond.

MS-DOS Hosted ROM Development Systems

Host + Target: \$750 Additional Targets: \$500

Targets:

- 6502 family
- 8080-8085-Z80-Z180-64180
- 8088-8086-80186-80286/8087-80287
- 68000-68010-68020/68881

Components:

- C compiler for host and target
- Assembler for host and target
- linker and librarian
- Unix utilities make, diff, grep
- Unix vi editor
- debugger
- download support

Features:

- Complete development system
- Fast development times
- Prototype and debug non-specific code under MS-DOS
- Compilers produce modifiable assembler output, support inline assembly, and will link with assembly modules
- Support for INTEL hex, S record, and other formats
- source for UNIX run time library
- processor dependent features
- source for startup

Aztec C Micro Systems

Aztec C is available for most micro-computers in three configurations: The Professional; The Developer; and The Commercial system. All systems are upgradable.

Aztec C68k/Am Amiga
source debugger—optional

Aztec C68k/Mac ... Macintosh
MPW and MAC II support

Aztec C86 MS-DOS
source debugger • CP/M libraries

The following have special pricing and configurations. Call for details.

Aztec C68k/At Atari ST

Aztec C80 CP/M-80

Aztec C65 Apple II & II GS

Standard System \$199

- C compiler
- Macro Assembler
- overlay linker with librarian
- debugger
- UNIX and other libraries
- utilities

Developer System \$299

- all Standard System features
- UNIX utilities make, diff, grep
- UNIX vi editor

Commercial System \$499

- all Developer features
- source for run time libraries
- one year of updates

MANX

C.O.D., VISA, MasterCard, American Express, wire (domestic and international), and terms are available. One and two day delivery available for all domestic and most international destinations.

Manx Software Systems
One Industrial Way
Eatontown, NJ 07724

CIRCLE NO. 104 ON READER SERVICE CARD

Aztec C is available on a thirty-day money back guarantee. Call now and find out why over 50,000 users give Aztec C one of the highest user-satisfaction ratings in the industry.

Call 1-800-221-0440

**In NJ or outside the USA,
call 201-542-2121**

Telex: 4995812 Fax 201-542-8386

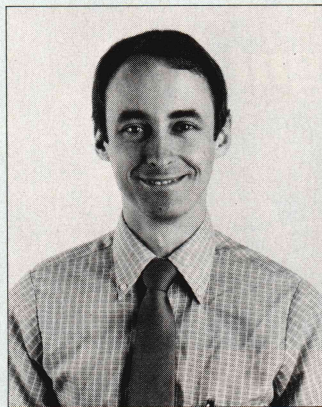
RUNNING LIGHT

First things first. Please note that the lack of a "C Chest" column this month doesn't mean we don't love C anymore. It's just that Allen Holub got involved with a few too many projects and had to take a vacation to decompress. Allen will be back next month, fully recovered. In the interim, this issue has plenty of C code to keep you busy.

Next topic: programming books. The floodgates have just started to crank open, and we're about to be inundated with books on programming for OS/2. Many of those books will be nothing more than boring regurgitations of the existing Microsoft documentation. Happily, I have seen some outstanding books lately, books written by programmers who really know what's going on inside OS/2, and I'd like to share the good news.

First up is Ed Iacobucci's *OS/2 Programmer's Guide* (Osborne McGraw-Hill, 1988). Iacobucci was the leader of the IBM OS/2 design team and clearly knows his stuff. This book is one of those (like Peter Norton's excellent *Programmer's Guide to the IBM PC*) that attempts to give you both the massive amounts of information that you really need and yet still be readable. At 1,100 pages, the book certainly is complete in coverage, and Iacobucci covers the material with a simple, clear style. This book is obviously the work of someone who's spent a lot of time helping other programmers get their programs running under OS/2, and wants to share his experience. Unlike most paperback computer books these days, this one's a steal at \$25.

Mere recommendations aren't really good enough for *Inside OS/2* by Gordon Letwin (Microsoft Press,



1988). Let me put it more directly: if you're at all serious about OS/2, you *must* buy this book. Letwin was the chief architect for Microsoft's OS/2 team, and with this book he's managed to pull off the nearly impossible trick of explaining the inner workings of

OS/2 along with the reasoning behind the architecture, while actually making the book fun to read. If you're new to OS/2, reading this book will take weeks off your learning curve. It isn't often that we're treated to a book this good.

Finally, if you're still writing code for MS-DOS instead gearing up for OS/2, I'm sorry to report that you're probably going to spend a great deal of money soon. The *MS-DOS Encyclopedia* is finally out, and it's generally a splendid (and massive) volume. Aside from the title, this book has nothing in common with the encyclopedia offered by Microsoft a few years back. This is a good reference, designed for programmers. There's coverage of the details of the file system, as well as articles on writing TSRs, device drivers and exception handlers, etc. There's a section on compatibility between DOS versions and OS/2, as well as developing for Windows.

My only problem with the book, besides its hefty price tag of \$135, is the dust jacket. The book was orchestrated and edited by friend and ex-DDJ columnist Ray Duncan, but somehow they forgot to put his name on the cover of the copy I received. The item they did remember is a nice bold credit for the foreword by Bill Gates. Funny how these things work out.

Tyler Sperry

Tyler Sperry
editor

ARCHIVES

Ten Years Ago Today

"...Pascal with extensions is a superior language for system programming, and we believe that it is in the public interest to assist in the current effort of many people and institutions to promote wider use of Pascal in place of some of the earlier high level languages. Though Pascal may have some shortcomings for specific applications when compared to specific proprietary languages, we regard it as by far the best general purpose language now in the public domain."—Kenneth L. Bowles "Status of the UCSD Pascal Project," DDJ, March 1978.

The object of this class is...

"Object-oriented programming began in the early 1960s with the development of Simula by the Norwegian Computation Center in Oslo, Norway. Simula was the first language to implement the Class construct. In the early 1970s, the Learning Research Group at Xerox Palo Alto Research Center began implementation of the Smalltalk programming environment. Smalltalk was the first system to be designed completely around the Class/Object concept. Many other languages have since provided support for classes, object, and subclassing, including CLU, Ada, C++ (an extended version of C), and several versions of LISP, though objects and classes are not as well integrated into these languages as they are in Smalltalk. The ease of experimentation with objects was one of the key reasons that much of the interesting user interface designs borrowed for Macintosh and Lisa were created in the Smalltalk system."—Bruce Horn, DDJ, October 1985.

DR. DOBB'S JOURNAL of
COMPUTER
Calisthenics & Orthodontia

Running Light Without Overbyte

New! Introducing Turbo C 1.5— the best optimizing compiler gets even better!

The professional optimizing compiler for less than \$100

Turbo C® is a technically superior production-quality compiler. (Borland's equation solver, Eureka™, is written in Turbo C.) And our Turbo C 1.5 offers a new library of the highest presentation-quality graphics in the industry—the kind you'll see in Quattro,™ our new professional spreadsheet.

And spectacular graphics are just part of the brand-new features. Turbo C 1.5 enhancements also include:

- A professional-quality graphics library of over 70 functions
- A librarian that allows you to build your own object module libraries
- Context-sensitive help for the language and the library routines



Actual photograph of Turbo C graphics displayed on IBM 8514 screen.*

- Text/video functions, including windows
- 43- and 50-line mode support
- VGA, CGA, EGA, Hercules, and IBM 8514 support
- File search utility (GREP)



- Sample graphics applications
- More than 100 new functions

For professional-quality C at an affordable price, no one else comes close to Turbo C. Because no one can deliver technical superiority like Borland.

60-Day Money-back Guarantee**

For the dealer nearest
you or to order, call
(800) 543-7543

Minimum system requirements: For the IBM PS/2™ and the IBM® and Compaq® families of personal computers and all 100% compatibles. PC-DOS (MS-DOS®) 2.0 or later. 384K.

*Artwork metallic courtesy of Genigraphics® Corporation

**Customer satisfaction is our main concern; it within 60 days of purchase this product does not perform in accordance with our claims, call our customer service department, and we will arrange a refund.

All Borland products are trademarks or registered trademarks of Borland International, Inc. Other brand and product names are trademarks or registered trademarks of their respective holders. Copyright ©1987 Borland International, Inc.

BI 11658

It's easy to upgrade to Turbo C 1.5!

Just complete this coupon and mail it with payment before June 30, 1988. Or, call us at (800) 543-7543 and be ready to give our operators your name, credit card number, and the serial number on your Turbo C master disk.

Turbo C 1.5 Upgrade Price

\$ 33.50

CA and MA residents add sales tax

Shipping and handling

In US \$5.00 (Outside US add \$10)

Total amount enclosed

\$

Must include your Turbo C serial

Return this coupon and the Turbo C RTL source code registration form from your Turbo C manual along with your payment by March 31, 1988 and receive your Turbo C 1.5 upgrade for free! (No phone orders please.)

Turbo C 1.5 Runtime Library Source Code

\$ 150.00

CA & MA residents add sales tax

Price includes shipping to all US cities.

(Outside US add \$10)

Total amount enclosed

\$

Please specify diskette size ☐ 5 1/4" ☐ 3 1/2"

Method of Payment: ☐ VISA ☐ MC ☐ Check ☐ Bank Draft

Credit card expiration date: ____ / ____

Card # _____

Name _____

Ship Address _____

City _____ State _____

Zip _____ Phone (____) _____

Mail coupon to: Turbo C 1.5 Upgrade Dept., Borland International
4585 Scotts Valley Drive, Scotts Valley, CA 95066

This offer is limited to one upgrade per valid product serial number. Not good with any other offer from Borland. Outside US make payments by bank draft payable in US dollars drawn on a US bank. CODs and purchase orders will not be accepted by Borland.

DDJ 3/88

BORLAND

Introducing a whole new world in C performance.

C6.0

WATCOM unleashes a high-performance optimizing C compiler and development system that delivers superior results for every professional programmer and offers the fastest programming environment you can get your hands on.

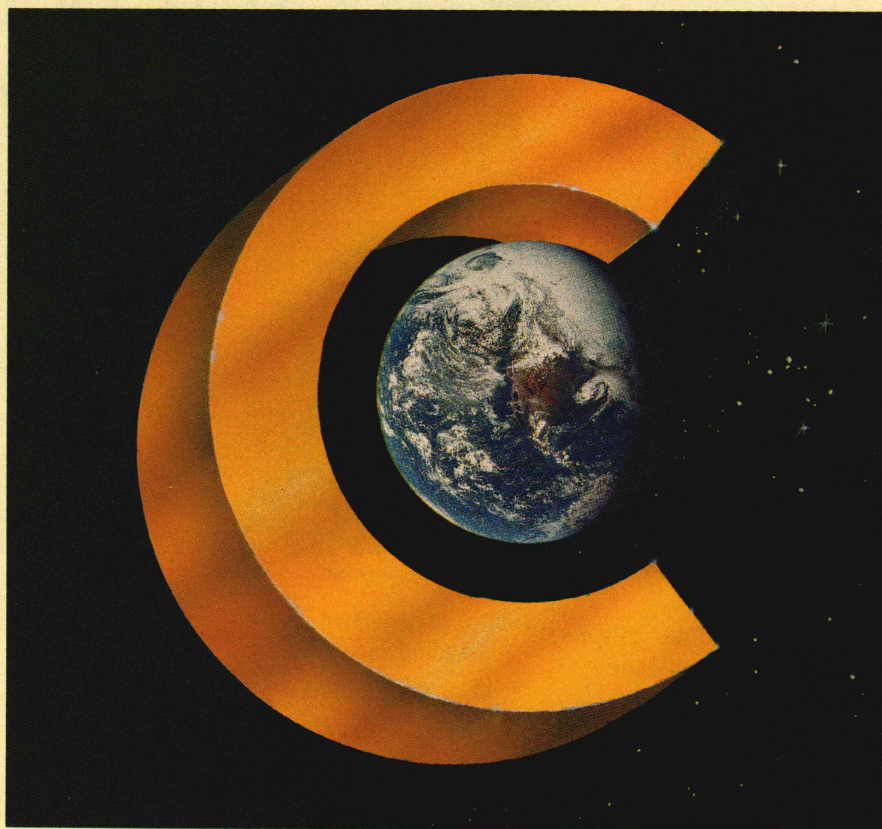
Producing the fastest, tightest code available today, WATCOM C6.0 offers major advantages in large memory models and floating-point computation.

The WATCOM C6.0 system includes WATCOM Express C (also sold separately), a seamless development environment complete with unexcelled diagnostic capabilities to maximize your productivity.

World Class Performance

Benchmark tests pitting WATCOM C6.0 against Microsoft C5.0 and Turbo C demonstrate the optimum code size and speed your programs will achieve with WATCOM C6.0. Sophisticated register allocation, true register variables and flow analysis allow your code to run its fastest. Moreover, extensive fine-tuning is possible with numerous user options like in-line substitution of machine code for performance-critical areas.

We deliver more than performance. Flexible run-time conventions allow efficient interfacing to a wide range of libraries and language processors. WATCOM C6.0 supports small, medium, compact, large and huge memory models and the NEAR, FAR and HUGE keywords.



Announcing the world's fastest, tightest code.

The fastest, tightest code
(small memory model*)

141	182	154
11.4	13.0	14.1
992	1246	1144
89.0	98.0	121.0

WATCOM C6.0 Microsoft C5.0 Turbo C

*IBM PC XT

25
Iterations
Sieve

Dhrystone

bytes
seconds

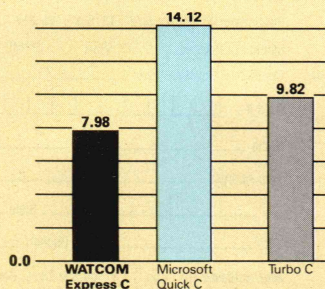
The fastest, tightest code
(medium memory model*)

150	184	159
2.7	3.0	3.4
1001	1232	1156
18.0	19.0	24.0

WATCOM C6.0 Microsoft C5.0 Turbo C

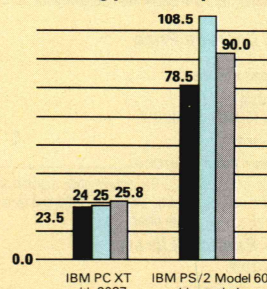
*IBM PS/2 Model 60

Turnaround Time (compile + link/load)



Time in seconds to compile 573 source lines plus includes, and link 7 additional OBJ files created from 1391 source lines. IBM PS/2 Model 60, small memory model. Program used: GREP.

Floating point computation



Time in seconds to run 25 iterations of Whetstone. Small memory model (64K code, 64K data).

Maximum Productivity

Express C's fastest available compile turn-around assists you with syntax and prototyping. With integrated tools like an editor and a symbolic debugger, you'll be amazed at how quickly you see your results.

The Right Tools for the Job

Express C, included with WATCOM C6.0, has unique error checking that quickly uncovers many common and difficult bugs in "correct" code that other compilers miss. Take advantage of the powerful integrated debugging in Express C, then use WATCOM C6.0 for even tighter code and faster execution speed. Comprehensive compile-time and run-time messages enable effective programming practices which you can extend with our fast development tools: **MAKE**, a **disassembler**, an object **librarian** and an overlay **linker**.

Technical Support

Because we know how valuable your time is, our on-line help is always available. Our software maintenance and licensing plans will keep you up to date. With them, our expert technical support is just a phone call away. When you create a great new product, our run-time library code can be licensed without royalties or accounting. Naturally, training seminars are available at our Customer Education Center or your site.

Superior Value

The WATCOM C6.0 system delivers superior value, returning your investment promptly by saving your valuable time. Our twenty years of computer language experience began in 1965 when our **WATFOR** authors wrote the book on high-productivity debugging compilers. The WATCOM C compiler is based on the sixth in a series of 8086-family code generators used since 1981. Today, we remain committed to delivering world class programming tools.

```
File Break Watch Colour Display
                                     (Ctrl)
AX=0000 BX=0094 CX=0001 DX=0567 SI=0C92 DI=0492 BP=0C76 SP=0C70
FL=010 P=0 A=0 Z=0 S=0 I=1 D=0 O=0 DS=43FE ES=43FE SS=424E IP=0029
SS:SP= 00EE BD33 0567 0000 0C92 0C92 0001 0C94 063F 0490 0492 0C90 0469
== Assembly ==
main+23      call    localtime
main+26      mov     -6(BP),AX          SS:0C70=000F
main+29      push    [01D2]
main+2D      mov     AX,0004          _WideTitle=0030
main+30      push    AX
== Source: cal ==
0033
0034      /* draw calendar for this month */
0035
0036      Calendar( time->tm_mon, time->tm_year, 10, 26, WIDE, WideTitle );
0037
0038      /* draw calendar for last month */
== Dialogue ==

temporary break point at main.cal(20)
(tm_sec=28, tm_min=1, tm_hour=1, tm_mday=9, tm_mon=8, tm_year=57, tm_wday=3,
tm_yday=156, tm_isdst=0)
Sun Mon Tue Wed Thu Fri Sat
DBG>
```

With WATCOM C6.0, you can debug large applications without extended memory.

WATCOM C6.0

Superlative Performance

- Full ANSI C Optimizing Compiler
- Source Editor
- Full-screen Source-level Debugger
- Full ANSI C Run-time Library
- Overlay Linker
- Object Librarian
- MAKE
- Disassembler
- **Express C**

List
price:
\$495

Introductory
Price:
\$295*

WATCOM Express C

Unparalleled Productivity

- Full ANSI C Compiler
- Integrated Source Editor
- Integrated Debugger
- Full ANSI C Run-time Library
- Integrated Linker/Loader
- Overlay Linker
- Object Librarian
- MAKE

List
price:
\$125

Introductory
Price:
\$75*

Available for IBM PCs, PS/2s and true compatibles with DOS. General availability March 1, 1988.

*Limited time introductory prices apply only to prepaid orders (VISA/MasterCard).
Shipping and handling extra.

WATCOM

_____ I want to order **WATCOM C6.0**,
regularly priced at \$495 and well worth it,
for just **\$295**.

_____ I want to order **WATCOM
Express C**, regularly priced at \$125 for
just **\$75**.

_____ Please send me organization site
licensing information.

_____ Please convince me further by
sending a complete product description of
WATCOM C6.0.

Name _____ Title _____

Company _____

Street _____

City _____ State _____ Zip _____

Information needed for Credit Card orders:

Signature _____ Tel. _____

Credit Card: Visa _____ MasterCard _____

Card # _____ Expiration Date _____

WATCOM
Dept. DD03B
1430 Massachusetts Avenue
Cambridge, MA 02138

**For even faster service call:
1-800-265-4555**

LETTERS

**Turbo C vs. Quick C**

Dear DDJ,

I read with great interest Allen Holub's article on C compilers in the October 1987 issue, in particular the comparison between Turbo C and Quick C. I largely agreed with his point of view about the programming environments, which I have little use for—they get in the way and are an additional source of compiler bugs. On other points, however, I disagree.

I finally received my Microsoft C 5.0 upgrade from 4.0 yesterday (December 4). Allen's complaints about the delays in delivery of Turbo C seem ironic to me, given the even longer delays in shipping Microsoft C and Quick C.

Allen's tests of the compilers must have been incomplete because he rates Quick C as clearly superior to Turbo C. There are several respects in which Turbo C 1.0 is superior to Quick C 1.0, and Turbo C increases its lead with Version 1.5.

First, Turbo C's documentation is much better. Second, Turbo C compiles 35 percent faster when you use Quick C's optimization switch. Third, even with Quick C's optimization turned on, executables produced by Turbo C run 15 percent faster than those generated by Quick C using my generic math function and numerical benchmark (*PC Tech Journal*, January 1988).

Fourth, and by far the worst, there is a serious bug in Quick C's numerical library: sines and tangents are apparently computed only to float accuracy, even when all real functions and variables are declared

double. Alone of seven PC C compilers I have tested, Quick C fails my double-precision accuracy diagnostic on these functions. Quick C's debugging facilities are superior to the nonexistent ones for Turbo C 1.0, but Borland says it will rectify this situation "soon."

The only reason I can see for using Quick C is to migrate easily to Microsoft C 5.0. Even so, nonoptimized compilation for Quick C takes almost 50 percent of that for Microsoft C 5.0. This doesn't seem worth it to me because Quick C doesn't produce the same numbers as Microsoft C 5.0.

Users of these compilers should be warned that optimization with Quick C does not change the numerical results (though it does markedly slow compilation), but iterative multiplications and divisions produce slightly different results with Microsoft C 5.0 when you optimize the loops.

Jim Roberts

3605 Centinela Ave., #2
Los Angeles, CA 90066

Allen replies:

I disagree.

First, release dates. There are two points:

1. Borland wouldn't give me a prerelease version of Turbo C. It gave a dog and pony show down in Scotts Valley but wouldn't give me a copy of the compiler to work with at home. I'm not willing to review a product that I can't work with on a day-to-day basis, and because I had a beta version of Quick C, I felt competent to talk about it, even though it hadn't been released at that time.

2. I agree, that neither company should have advertised their products as finished until they were. The problem here is projected release dates vs. advertising deadlines. You have to place an ad several months in advance of its publication, and it is usually impossible to guess accurately when a real product will be finished. I think that both Microsoft and Borland jumped the gun, however.

Second, by my standards Turbo

C's documentation is inferior to Quick C's. The two users' guides are comparable, but Turbo C's library documentation is miserable. It's poorly organized and formatted; the programming examples—if they're there at all—are trivial; and obscure and nonstandard subroutines are often documented (and I use the word loosely) in only two or three sentences. I use the run-time library documentation for my compiler on a daily basis; I read the users' guide once when I get the compiler. Consequently, the quality of the former is much more important, and Turbo C falls down in that department.

Next, I'm not sure about the comparative compile times with the newest Turbo C version. I haven't seen a copy. As I said in the review, I was using Turbo C 1.0 and a beta version of Quick C, so it's likely that things have changed. Mr. Roberts and I are testing the compilers in radically different ways, however. My benchmarks were two very large programs (versions of the Unix lex and yacc utilities) that used virtually no numeric computations but did a lot of data manipulation, especially at the bit level. Mr. Roberts seems to be compiling smaller, numeric-intensive programs.

Because I disliked both of the interactive environments, I compiled with the command-line versions of both compilers, the compile process driven from a make file. I did find that it took longer to load Quick C than it did to load Turbo C. In fact, on a short program, it often took more time to load Quick C than to run it. Because I was compiling relatively large modules, however, the load time was insignificant when measured against total compile time.

Compile time is relatively insignificant in the greater scheme of things, however. If I save five hours with a good debugger but waste an hour with longer compile times, I'm still four hours ahead. Quick C's debugging facilities are indeed superior to the nonexistent ones for Turbo C, and the debugger alone makes Quick C a considerably more useful compiler, especially if you're just learning the language. Borland



#1 PROGRAMMABLE EDITOR

Call 1-800-45-VEDIT for
FREE Fully Functional Demo Disk

Stunning speed. Unmatched performance. Total flexibility. Simple and intuitive operation. The newest VEDIT PLUS easily satisfies the most demanding computer professional.

Try a Dazzling Demo Yourself.

The free demo disk is fully functional—you can try all features yourself. Best, the demo includes a dazzling menu-driven tutorial —you experiment in one window while another gives instructions.

The powerful "macro" programming language helps you eliminate repetitive editing tasks. The impressive demo/tutorial is written entirely as a "macro"—it shows that no other editor's "macro" language even comes close. And VEDIT PLUS is only 40K in size.

Go ahead. Call for your free demo today. You'll see why VEDIT PLUS has been the #1 choice of programmers, writers and engineers since 1980.

Only VEDIT PLUS is this Flexible.

The installation lets you pick from closely emulating the keyboard layout of Word Perfect, WordStar and others. Or you can easily create your own layout and even your own editing functions. Supports any screen size—you pick screen colors and attributes.

Supports the IBM PC, XT, AT and PS/2. Also supports MultiLink, PC-MOS/386, Concurrent DOS and most networks. Also available for MS-DOS, FlexOS (protected mode), CP/M-86 and CP/M. (Yes, we support windows on most CRT terminals, including CRTs connected to an IBM PC.) Order direct or from your dealer. \$185.

Special: VEDIT (single file, no windows) for CP/M—\$49.

- Fully Network Compatible
- Call for XENIX-286 version
- 30 Day Money-back guarantee

Compare Features and Speed

	BRIEF	Norton Editor	PMATE	VEDIT PLUS
'Off the cuff' macros	No	No	Yes	Yes
Built-in macros	Yes	No	Yes	Yes
Keystroke macros	Only 1	No	No	Unlimited
Multiple file editing	20 +	2	No	20 +
Windows	20 +	2	No	20 +
Macro execution window	No	No	No	Yes
Pop-up menus	No	No	No	Yes
Execute DOS commands	Yes	Yes	Yes	Yes
Automatic processing of				
Compiler errors	Yes	No	No	Yes
"Cut and paste" buffers	1	1	1	36
Undo line changes	Yes	No	No	Yes
Paragraph justification	No	No	No	Yes
Convert to/from WordStar	No	No	No	Yes
On-line calculator	No	No	No	Yes
Configurable Keyboard	Hard	No	Hard	Easy
43 line EGA support	Yes	No	No	Yes
Manual size/index	250/No	42/no	469/Yes	380/Yes
Benchmarks in 120K File:				
2000 replacements	1:15 min	34 sec	1:07 min	6 sec
Pattern matching search	20 sec	Cannot	Cannot	2 sec
Pattern matching replace	2:40 min	Cannot	Cannot	11 sec



VEDIT and CompuView are registered trademarks of CompuView Products, Inc. BRIEF is a trademark of UnderWare, Inc. PMATE is a trademark of Phoenix Technologies Ltd. Norton Editor is a trademark of Peter Norton Computing Inc. MultiLink and PC-MOS/386 are trademarks of The Software Link, Inc. CP/M and FlexOS are trademarks of Digital Research. MS-DOS is a trademark of Microsoft.

*Also available for TI Professional, Tandy 2000, DEC Rainbow, Wyse WY700 and others.
*Demo disk is fully functional, but does not readily write large files.

CIRCLE NO. 107 ON READER SERVICE CARD

CompuView

1955 Pauline Blvd., Ann Arbor, MI 48103
(313) 996-1299, TELEX 701821

seems to be fostering rumors that it's working on a debugger, but the last time I talked to the people at the company, they wouldn't even commit to the fact that a debugger was in progress, much less give me a projected release date. It may be available "soon," but are they talking geologic time or historical time?

Other problems are reliability and portability. My Turbo C-compiled benchmark didn't run initially because of bugs in and omissions from the I/O library. There was no `signal()` function, `getenv()` didn't work properly (it returned a `PATH` environment when I had asked for a `PA` environment but had set `PATH` first), and so forth.

Of course, by the time you read this letter, things may have changed: Borland may have fixed the documentation, improved the library, and come out with a debugger to match dbx on a Sun workstation. But I'm only willing to discuss products that I actually have in my hands. Similarly, I test compilers by working with them rather than by running

artificial benchmarks. Consequently, I'm likely to miss a few things that I don't use very much (like the accuracy of a sine function), but I think my methods give a much better feel for how a product works in general than do more formal benchmarks.

Dear DDJ,

We'd like to set the record straight on the shipment of Borland International's Turbo C.

Borland announced shipment of Turbo C and distributed the product on the same day: May 14, 1987. We said we'd ship in the second quarter of 1987, and we made it by over a month and a half.

We'd also like to point out that Borland collected no money before the product began shipping. As a matter of policy, Borland neither cashes checks nor bills credit cards until a product is shipped.

With the volume and depth of information that Dr. Dobbs' offers, it's hard to check every detail. Thanks for letting us contribute to your fine publication.

Borland International

The editor replies:

It was never our intention to imply that the folks at Borland were committing mail fraud, or anything of the sort. Borland has generally performed very well in customer relations.

It was incongruous, however, to have Turbo C arrive at our offices with a note from Philippe Kahn proclaiming the product was shipping "right on time"—three months after the first ads had appeared in magazines. To claim that an advertisement with a snip-out coupon and an 800 number for credit card orders isn't an official product announcement is mere sophistry; ads are designed with coupons and credit card phone numbers to generate an immediate response. Our readers had every reason to expect shipment within 30 days of the first ad.

In short, if Borland had originally planned to ship Turbo C in the second quarter it would have been more honest to either postpone the ad a few months or include a disclaimer ("please allow 3 to 4 months for shipment").

Corrections

Dear DDJ,

Readers, your response to "An Extended IBM PC COM Port Driver" in the June 1987 issue of DDJ was gratifying. The version of `excom.asm` available from DDJ on diskette seems to be working, whereas `exmode.c` has a serious bug. `Exmode` will work correctly if you have two COM ports and will not do anything right if you only have one port. Specifically, in one-port systems `exmode` will always respond with "excom not installed" and fail to execute your commands. To correct this, change the `&&` in the first `if` statement of `main` to `!`. In addition, some early versions of `exmode` used `remove` as a function name. This conflicts with some libraries and can be corrected by renaming the function. How embarrassing, sorry 'bout that.

Tom Zimniewicz
Lima, NY 14485

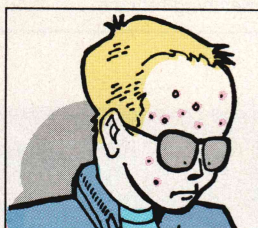
DDJ

NINE KINDS OF PROGRAMMERS:

We're not **just** nerds, anymore!



consultant



developer



hacker



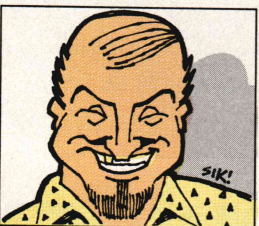
hobbyist



lifer



pirate



profiteer



researcher



unemployed

PVCS

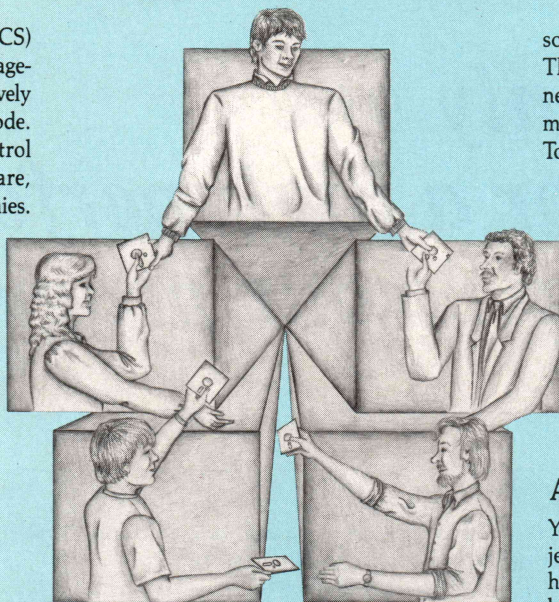


The Number One Source Code Control System.

The POLYTRON Version Control System (PVCS) simplifies and automates Configuration Management so programmers and managers can effectively control the revisions and versions of source code. PVCS is the most widely used change control product and is used by the leading software, aerospace, manufacturing and service companies.

"In terms of features, PVCS provides everything necessary to a large multi-programmer project — more than any other package reviewed. No restrictions are placed in the development environment and all aspects of operation can be customized for specific project needs."

PC Tech Journal
September 1987



source files, libraries, object code and other files. The levels of security can be tailored to meet the needs of nearly every project. PVCS works on all major LANs including 3Com, Novell and the IBM Token Ring Network.

"PVCS has helped us maintain nearly 90 programs and utilities. Without it we would not have the quality of our upcoming release of NetWare."

Jonathan Richey
Manager, NetWare Utilities
Novell

Unmatched Flexibility

- Storage & Retrieval of Multiple Revisions of Source Code
- Maintenance of a Complete History of Changes
- Control of Separate Lines of Development (Branching)
- Resolution of Access Conflicts
- Optional Merging of Simultaneous Changes
- Release and Configuration Control
- Project Activity Reports
- Management Reports
- Command or Menu Interface

Project Control

PVCS maintains individual archives of all project components in your system — source code modules, data files, documentation and even object code libraries. These "source documents" can be written in any language or multiple languages.

Fast Retrieval of Revisions

PVCS uses "reverse delta storage" which saves disk space and speeds retrieval of versions of any file in the project database. A delta is the set of differences between any revision and the previous revision. PVCS can rapidly recreate complete versions of any file whether it is the most recent revision of a module or the original version of the entire project. Differences are automatically detected and stored.

A Practical Necessity for LANs

While important for single-programmer projects, PVCS is absolutely essential for multiple-programmer projects and LAN-based development efforts. In a LAN environment, source code files are simply too easy to change. Because any change to any file can have major ramifications, coordinating and keeping a record of changes is critical. Project leaders can determine, on a module-by-module basis, which programmers can access or modify

Once you standardize on PVCS, the archives used to track and monitor changes are interchangeable between any PVCS product.

Personal PVCS — Offers most of the power and flexibility of Corporate PVCS, but excludes the features necessary for multiple-programmer projects.

Corporate PVCS — Offers additional features to maintain source code of very large and complex projects that may involve multiple programmers. Includes multi-level branching to effectively maintain code when programs evolve on multiple paths.

Network PVCS — Extends Corporate PVCS for use on Networks. File locking and security levels can be tailored for each project.

PVCS for VAX systems — Requires VMS. Uses the same interface and archive format as MS-DOS version. Supports branching and offers file locking and other security features for multiple-programmer projects.

Adopt PVCS on Your Existing Projects

You can obtain the benefits for your current project without disrupting development, regardless of how long your project has been under way. You can build PVCS archives from revisions stored in your present files or simply adopt PVCS from the current date.

PolyMake Reads PVCS Logfile Format

PolyMake, the leading Make utility, understands the structure of PVCS logfiles and is able to correctly determine the date and time of any revision. This prevents unnecessary operations that occur when the date and time of the complete project archive itself is used as with other make utilities.

	MS-DOS*	VMS		
	PC/XT/AT	Micro VAX II	VAX 7xx	VAX 8xxx
Personal PVCS	\$149			
Corporate PVCS	\$395			
Network PVCS	\$995**	\$4,950	\$9,500	\$10,500+
PolyMake	\$149			
Network PolyMake	\$447**	\$1,250	\$2,375	\$2,500+

**5 Station LAN License. Call for pricing on larger Networks.

TO ORDER:
1-800-547-4000
Dept. DDJ

Oregon & Outside USA call (503) 645-1150.
Send Checks, P.O.s to: POLYTRON
Corporation, 1700 NW 167th Place,
Beaverton, OR 97006



High Quality Software Since 1982

CIRCLE NO. 108 ON READER SERVICE CARD

*There is only one
true test for a high performance
C compiler....your program*

```
foval = segs[mp->mod_segidx[target_datum]].seg_offset
+ mp->mod_segoff[target_datum];
foval += target_displacement;
foval += (tframe = segs[mp->mod_segidx[target_datum]].seg_frame) << 4L
- ;
/* abs loc seg */
if (!(segs[mp->mod_segidx[target_datum]].seg_acbp & 0xe0))
    no_base = 1;
break;
case 1: /* group index */
    gpctr = grps[mp->mod_groups[target_datum]]->sym_grp;
    foval = segs[*gpctr->grp_segarr].seg_offset;
    foval += (tframe = gpctr->grp_frame) << 4L;
    foval += target_displacement;
    break;
```


The most important measurement for rating your C compiler's execution speed is the performance of your program. C86PLUS' multi-pass optimizer concentrates on value use analysis and register selection, as opposed to areas that have little or no impact on program size or speed except in benchmarks.

With this approach in mind, Computer Innovations' C86PLUS v.1.10 is geared for large, real world applications rather than promotional benchmarks, which in many cases, do not accurately reflect superior code generation.

And while most compiler vendors rely on their benchmark claims and lengthy optimization listings to sell speed, we're relying on what really counts the most -- the performance of your program.

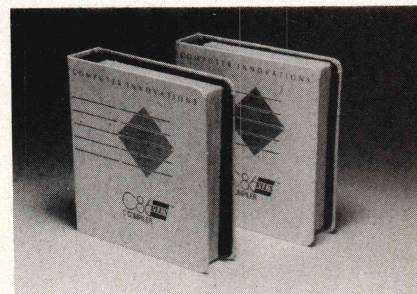
C86PLUS has the features you need for professional-level application development. The latest proposed ANSI C libraries; 80386-specific compiler options to take advantage of the 32-bit architecture while running under DOS in real mode; 100%

ROMable code for embedded systems development; a MAKE utility; C library source code at no extra charge; and FORTRAN, Pascal calls.

Small, medium, compact and large memory model support; powerful math libraries and inline floating point support for computation-intensive programming; Microsoft C v.4.0 and UNIX System V compatibility for application portability; powerful support tools for increased programmer productivity; and the ability to write interrupt routines in C rather than assembler.

Support to keep you satisfied. Whatever the time or problem, we're prepared to help. Computer Innovations (CI) offers telephone technical support during regular business hours, a dial-up 24-hr. bulletin board service or you can contact us via our vendor conference on BIX.

Should add-on productivity products be of concern, C86PLUS is supported by most major third-party vendors and CI resells them for your added convenience. CI also offers products specific to embedded systems application development and C language training.



Experience performance from a real point-of-view. Order now or contact us for more information.

800-922-0169

**COMPUTER
INNOVATIONS**

**980 SHREWSBURY AVE.
TINTON FALLS, NJ 07724, USA
TLX: 705127 COMP INNOV UD
(201) 542-5920**

C86PLUS[®]
C COMPILER
(call for current pricing and version)

Minimum System Configuration: Intel 8086/186/286/386-based system running DOS 2.1 or above; 384K RAM and a hard disk drive is required. Registered trademarks: Microsoft (Microsoft Corp.), UNIX (AT&T-Bell Labs), Intel (Intel Corp.), BIX (McGraw-Hill, Inc.).

CIRCLE NO. 109 ON READER SERVICE CARD

Object-Oriented Programming and Databases

by Jacob Stein

Is structured programming still relevant? In the early 1970s, E.W. Dijkstra introduced us to structured programming and Nicklaus Wirth introduced us to stepwise refinement. These programming methodologies were intended to foster a systematic and scientific approach to software development. That they were an improvement over previous methodologies, or the lack thereof, is not in doubt. What is in doubt is whether they are still germane to many applications being developed today. Structured programming appears to fall apart when applications exceed 100,000 or so lines of code. Programmer productivity fails to meet expectations and software maintenance becomes a nightmare. What we are left with is programmer fear: fear of side effects when fixing even minor bugs; fear of improving what already works, even if it does so poorly.

It may be that the complexity of the application itself is responsible for this failure. On the other hand, our understanding of the application domain may be the limiting factor. Although the underlying models for banking and other conventional applications have been understood for millenia, the underlying models for new application areas such as computer-aided software engineering (CASE) and computer-integrated manufacturing (CIM) are as much a product of the process of applica-

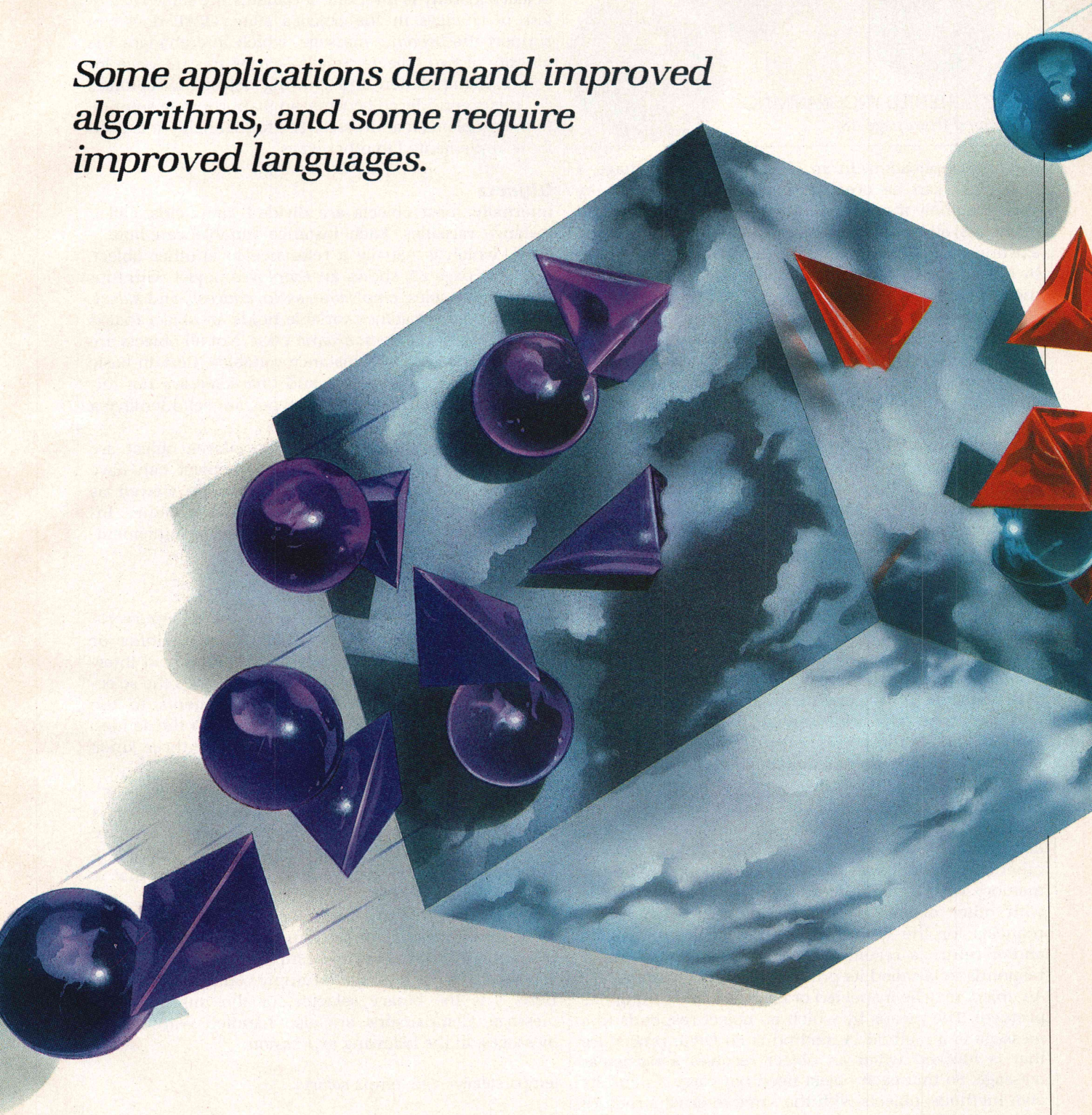
tion development as they are inputs to the process. Structured programming assumes that the application developer stands a reasonable chance of getting it right the first time, and that once the application has been successfully coded, the application domain will not change. For many applications being addressed today, neither of these is likely.

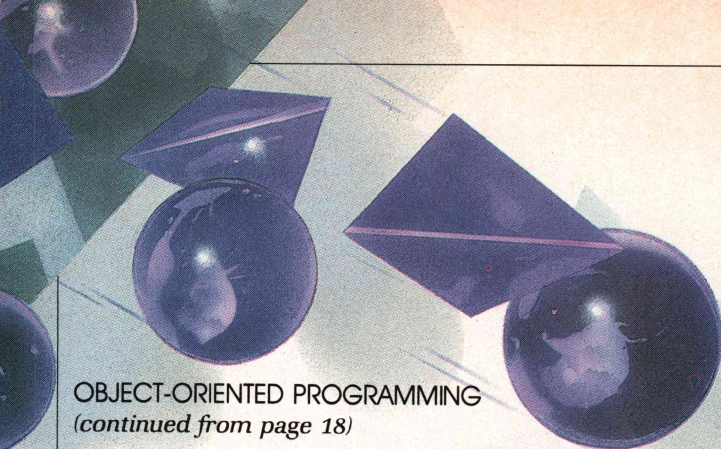
Object-oriented programming provides a paradigm suitable for these new application areas. Instead of looking at programs as algorithms and data structures, it looks at applications as the behavior of objects and the communication between objects. This communication takes place in the form of messages to which objects respond with suitable behavior. In addition, object-oriented programming addresses the vital need to validate abstractions before committing many man-years to a detailed implementation and the fact that application domains change with time. Diedrich and Milton refer to the style of development encouraged by object-oriented languages as "fearless programming."¹

By organizing objects into classes, which encapsulate object behavior in a manner similar to abstract data types, and organizing classes into an inheritance hierarchy, object-oriented programming may finally provide the long sought after sharing and reusability of code. The code that gives a data structure its meaning, its semantics, is no longer spread amongst disparate application programs. Perhaps one of the most frustrating aspects of trying to reuse code in a new application is that the old code almost, but not quite, fits the needs of

Dr. Jacob Stein has been a senior software engineer at Servio Logic Corp. of Beaverton, OR since 1984. He is also an adjunct assistant professor at the Oregon Graduate Center.

Some applications demand improved algorithms, and some require improved languages.





OBJECT-ORIENTED PROGRAMMING

(continued from page 18)

the new application. In an object-oriented language, a new class can be created from an existing class by specifying only the differences between the existing and new class; all else is inherited by the new class from the existing class. PPI's belief in the encapsulation provided by object-oriented systems and lead them to trademark the term *Software-IC*.

Object-oriented languages tend to be memory-based and are geared toward supporting a single user or application. Objects either do not persist between program executions or are saved in a primitive manner that does not lend itself to sharing (for example, copying a program's image to disk). It would be difficult for a design layout tool and a design rule checker to share data, let alone run concurrently. Object-oriented data management systems rectify this situation by providing data management facilities. The trick here is to provide these facilities in a manner that provides flexibility and merges well with the object-oriented paradigm.

An Object-Oriented Model

This section outlines the OPAL programming language and GemStone data model developed by Servio Logic Corp. Most of the constructs described here are present, in one form or another, in all object-oriented languages. The concepts of objects, messages, methods, classes, and class hierarchy are essential to all object-oriented models. You should beware of systems that claim to be object-oriented yet do not contain similar concepts.

The three principal concepts of the model and language are object, message, and class. They correspond roughly to record, procedure call, and record type in conventional systems. (Table 1, page 22, gives some other correspondences.) An object is a chunk of private memory with a public interface. Objects communicate with other objects by passing messages, which are requests for the receiving object to change its state and/or return a result. The set of messages an object responds to is called its protocol (its "public interface"). An object may be inspected or changed only through its protocol. The means by which an object responds to a message is a method. A method is an OPAL procedure that is invoked when an object receives a particular message. So that each object need not carry around its own methods, objects with the same internal structure and methods are grouped into a class and are called instances of the class. The methods and structure are in a single object describing the class—the class-defining object (CDO)—and all instances of the class contain a

reference to the class-defining object. Unlike some other object models, an object is an instance of exactly one class.

Every object has a unique identity. In OPAL, an object is identified by its object-oriented pointer (OOP). An object's identity is invariant: it remains the same regardless of changes in the object's state. OPAL does not support the *become*: message, which interchanges the identity of two objects. Smalltalk uses this message for screen manipulation and growing objects. GemStone's designers considered *become*: an inappropriate solution to the problems it addresses. The functionality it is used for is implemented in other ways.

Objects

Internally, most objects are divided into fields, called instance variables. Each instance variable can hold a value, which is usually a reference to another object. Figure 1, page 22, shows an *Employee* object with four instance variables: *empName*, *ssNo*, *address*, and *salary*. The *empName* instance variable holds an object that is an instance of the *PersonName* class. Not all objects are internally divided into instance variables. Certain basic types such as *SmallInteger* and *Character* are not further decomposed. These basic types are self-identifying as they have a direct representation.

The instance variables in the *Employee* object are called named instance variables. An object can have indexed instance variables, which can be viewed as instance variables with numbers instead of names. Indexed instance variables are used mainly for implementing ordered collections, such as arrays.

Messages

The basic form of all message expressions is *<receiver>* *<message>*. The *<receiver>* part is an identifier or expression denoting an object that receives and interprets the message. The *<message>* part gives the selector of the message and possibly arguments to the message. Every message returns a result to the sender, which is usually another object. There are three kinds of messages: unary, binary, and keyword.

Unary messages have no arguments and have selectors that are a single identifier. Assume that *emp* is a variable holding an *Employee* object. If there is a unary message *firstName* to retrieve the first name of the employee, then *emp firstName* returns a string that is the first name of *emp*.

Binary message expressions have a receiver, one argument, and a message selector that is one or two nonalphanumeric characters. To multiply 8 by 3, for example, you send to 8 the message "Multiply yourself by 3": *8 * 3*. Here, *** is the binary selector for the multiplication message. Comparisons are also handled with binary messages. In the following expression:

```
(emp1 salary) <= (emp2 salary)
```

the receiver and argument of the *<=* binary message are both results of unary message expressions.

Keyword messages have one or more arguments and have multipart selectors composed of alphanumeric

Break the 640K Barrier for \$59.95.

Introducing Quarterdeck Expanded Memory Managers™: QEMM 386 and QEMM 50/60.

Your 80386 PC, IBM® Personal System/2™ Model 80, PC or AT with 80386 add-in board, as well as your IBM Personal System/2 Models 50 or 60 can all break through the DOS 640K barrier. Now you can have maximum use of your memory—whether you have one megabyte or 32— with the Quarterdeck Expanded Memory Manager. All without having to purchase special expanded memory boards.

Unlock hidden potential beyond 640K.

QEMM uses hidden features within your existing memory to make it compatible with the Lotus-Intel-Microsoft Expanded Memory Specification (EMS) version 4.0.

Now you can run colossal spreadsheets, databases, and CAD models designed for expanded memory. Using Lotus 1-2-3, Symphony, Framework, Paradox, AutoCAD, Microsoft Excel and more.

Improve productivity with DOS multitasking.

And if you'd like to use these programs all together

—multitasking beyond 640K— QEMM extends the capabilities of our popular DESQview™ multitasking environment.



DESQview lets your programs work together in a familiar way, giving you many of the capabilities and features only promised by other systems. In fact, the editors of *PC Magazine* recently named it "Best Alternative to OS/2," and *InfoWorld* gave it a report card rating of 9.1.

We offer productivity solutions for the forgotten 12 million.

If you are one of the 12 million or so 8088, 8086 or 80286 PC users who feel left out of this new world, don't despair. We have options that let you keep your computer and favorite programs and give you today what the newest computers and operating systems are promising for the future.

Visit your dealer for more information on Quarterdeck products. And ask to see the chart we've prepared explaining today's maze of memory options. and how you can break the 640K barrier.

For 386 PC, IBM PS/2 model 80 and 80386 add-in board users, QEMM/386:

- Fully supports the Lotus-Intel-Microsoft Expanded Memory Specification (EMS) version 4.0
- Enables the 80386 user to configure the 80386's memory beyond the first megabyte as expanded memory or as a combination of both expanded and extended memory.
- Fills out missing DOS memory to 640K and beyond by mapping into unused video memory. When used with a CGA graphics adapter this can give you 736K instead of the standard 640K for DOS.
- Automatically detects the speed of the memory in your PC and uses fast memory whenever possible.
- Increases system performance by remapping slow ROM into the 386's fast RAM memory.
- Frees up DOS conventional memory by loading memory-resident programs above video buffers.
- Supports 386 protected-mode programs, such as Ansa's Paradox 386, that use Phar Lap Software's 386 DOS Extender.
- Turns DESQview into a 80386 control program, using the 80386's 8086 machine architecture to run multiple large programs concurrently.
- Takes approximately 1.5K of overhead from the PC's conventional memory used to run DOS programs.

System Requirements: 80386 PC, IBM PS/2 Model 80, or standard 8088, 8086, 80286 with 80386 add-in board.

For IBM PS/2 model 50 and 60 users, QEMM-50/60:

- Fully supports the Lotus-Intel-Microsoft Expanded Memory Specification (EMS) version 4.0.
- Takes advantage of the hardware capabilities of the IBM PS/2 80286 Memory Expansion Option to transform its "extended" memory into "expanded" memory.
- On PS/2 Model 50 and 60's with 1.5 MB or more of the IBM Memory Expansion Option memory, enables DESQview to run multiple large programs concurrently.

System Requirements: IBM PS/2 model 50 or 60 with IBM PS/2 80286 Memory Expansion Option board or compatible. Note: IBM PS/2 architecture requires disabling motherboard memory and 'backfilling' conventional memory to 1 MB in order for any multitasking environment to work, including Microsoft Windows 2.0, IBM's 3270 workstation program, or DESQview. QEMM will not run on 80286 computers with extended memory.

Rush Me Quarterdeck Productivity Solutions Today!

DDJ 3/88

Product	No. of Copies	Media 3 1/2" or 5 1/4"	Retail Price, ea.	Total
QEMM/386			\$59.95	
QEMM 50/60			\$59.95	
DESQview 2.0			\$129.95	
Shipping & Handling		USA	\$ 5.00	\$
Payment method: <input type="checkbox"/> Check		Outside USA	\$10.00	\$
<input type="checkbox"/> VISA <input type="checkbox"/> MC <input type="checkbox"/> AMEX		Sales Tax (CA residents)	x 6.5%	\$
Valid Since _____ / _____		Expiration _____ / _____	Amount Enclosed	\$

Card Number:

Name on Credit Card: _____

Shipping Address: _____

City: _____

State: _____ Zip: _____ Tel: _____

Mail to address at right.

Note: If you own DESQview call us for a special upgrade offer, or send in your DESQview registration card. AST Special Edition owners included.

Quarterdeck
QEMM and DESQview
Quarterdeck Office Systems, 150 Pico Blvd.
Santa Monica, CA 90405 (213) 392-9851

characters and colons. To set the third component of an array held in *anArray* to 'Ross', you use:

```
anArray at: 3 put: 'Ross'
```

The same effect is accomplished in other languages by:

```
anArray[3] := 'Ross'
```

The message selector here has two parts—*at:* and *put:*—because it takes two arguments: the array index and the object to be stored at that index. You refer to a message as referred to by the concatenation of its parts: *at:put*.

Methods

To construct a method you need to know what objects are visible within its scope. All the named instance variables of the receiver are available via their names. Thus, in a method for class *Employee*, as defined in Figure 1, the instance variables *empName*, *ssNo*, *address*, and *salary* are accessible.

A method may also have temporary variables, which are declared between vertical bars at the beginning of the method: *temp1 temp2*. Each user has one or more dictionaries of global variables, and those global variables can appear in methods. You need two other operators before you can write methods: the assignment operator (*:=*) and the operator *^*, which returns the value of the expression as the result of a method.

For a unary message *wholeName* in class *PersonName* that returns the first and last name concatenated with a space between, you can use the following method:

GemStone	Conventional
object	record instance, set instance
instance variable	field, attribute
instance variable constraint	field type, domain
message	procedure call
method	procedure body
class-defining object	record type, relation scheme
class hierarchy	database scheme
class instance	record instance tuple
collection class	set, relation

Table 1: Correspondences between concepts in GemStone and conventional systems

```
wholeName
```

```
temp1
temp := first.
temp := temp + ' ' + last.
^ temp
```

The first statement, *temp := first*, assigns the value of the instance variable *first* in the receiver (a *PersonName* object) to the temporary variable *temp*. The statement:

```
temp := temp + ' ' + last
```

concatenates the value of *temp*, a blank, and the contents of the instance variable *last* of the receiver (+ is the binary message for concatenation of strings). The result of the concatenations is assigned to *temp*. Finally, the statement *^temp* returns the value of *temp* as the result of the message *wholeName*. (This particular message could be implemented with the single message expression *^first + ' ' + last* with no need for a temporary variable.)

A method can also change the state of the receiver, by

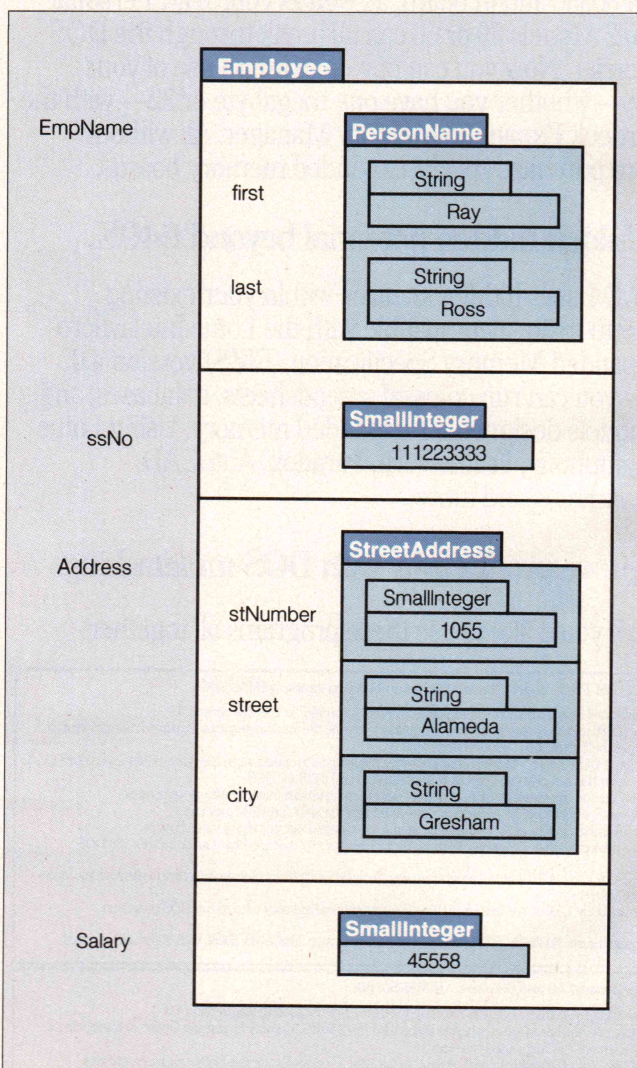
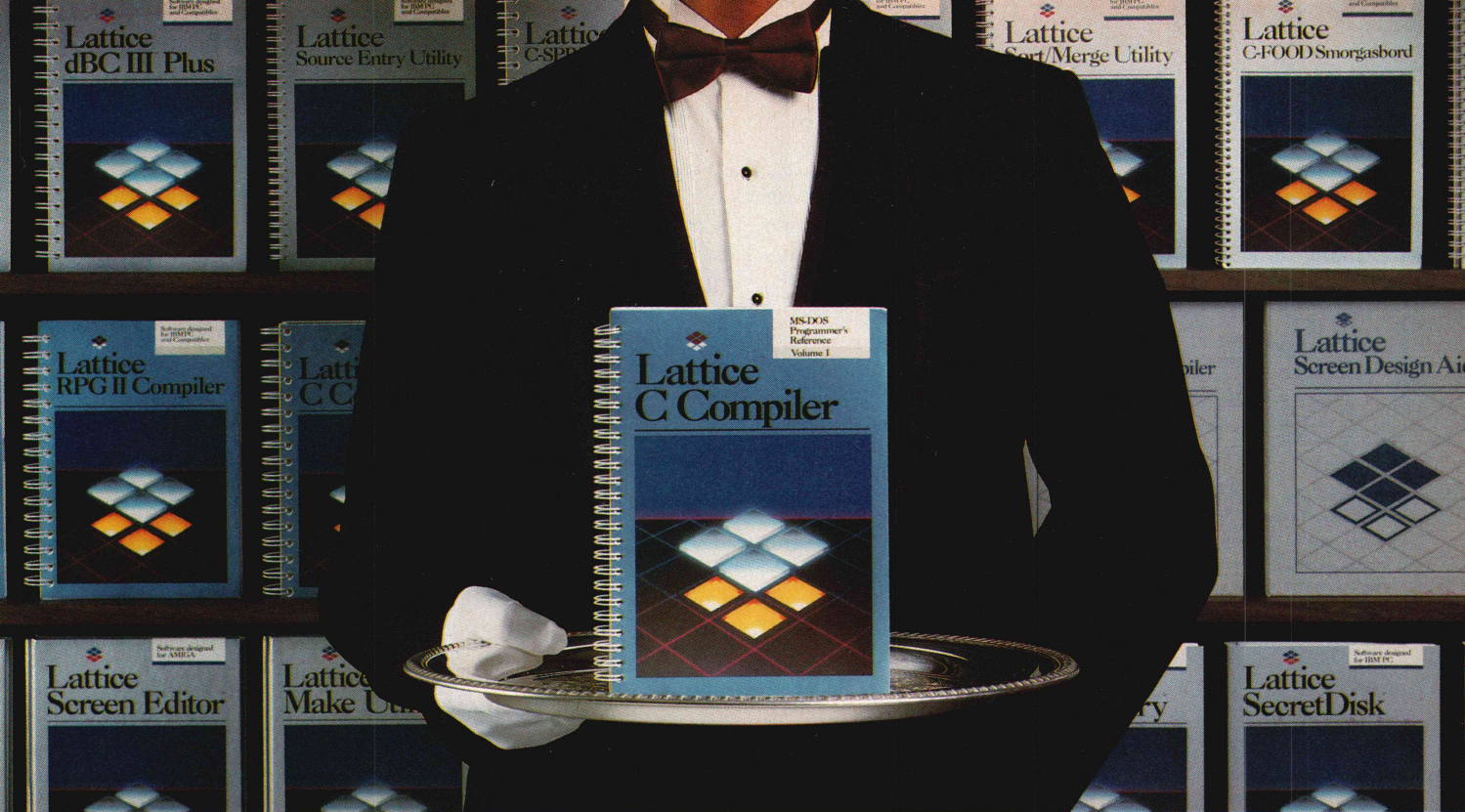


Figure 1: An Employee object and its four instance variables



Our software comes with something no one else can offer.

When you join the Lattice family of customers, you'll discover that your software purchase is backed by more than just an excellent warranty. It's backed by unparalleled technical support. By a total commitment to your success and satisfaction. And by Lattice's dedication to excellence in products and services.

Unlike other software manufacturers who charge you for services after you've purchased their product, Lattice offers a unique package of support programs at a price we can all live with—FREE.

Lattice Bulletin Board Service
LBBS is our 24-hour a day bulletin board system that allows you to obtain notification of new releases, general information on Lattice products, and programs for the serious user. And if you've ever experienced the frustration of having to wait a year or more for a new release (that has corrected a bug), you'll really appreciate LBBS. Because with this service, you can actually download the latest program fixes to instantly eliminate any bugs discovered after release.

Available through dealers and distributors worldwide.

Lattice Service.

Technical Support Hotline

Responsible, dependable and capable Support Representatives are only a phone call away. You will talk to a highly skilled expert who is trained to answer any questions you have relating to specific Lattice products. Remember, your complete satisfaction is our goal.

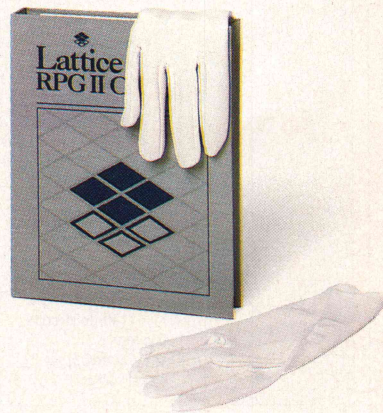
McGraw-Hill BIX™ Network

The Byte Information Exchange (BIX) Network is a dial-in conference system that connects you with a Special Interest Group of Lattice users. The nominal one-time registration fee allows you to BIX-mail your questions—via your modem—directly to Lattice. Or you can post your questions in the conference mode for Lattice or other users to answer. Once again, you have 24-hour access.

You Also Receive:

- Timely updates and exciting enhancements
- 30-day, money-back guarantee
- Lattice Works Newsletter
- Technical Bulletins
- Access to Lattice User Groups

Lattice has developed more than 50 different Microcomputer software tools that are used by programmers worldwide. We were there for every MS-DOS release. We're there now for OS/2. And we'll be there for the next generation of technical changes. But most of all, Lattice is there for you.




Lattice

Subsidiary of SAS Institute Inc.

Lattice, Incorporated
2500 S. Highland Avenue
Lombard, IL 60148
Phone: 800/533-3577
In Illinois: 312/916-1600

CIRCLE NO. 111 ON READER SERVICE CARD

assigning new values to instance variables. Suppose objects of class *Department* have a *budget* instance variable. The following method increases a department's budget by a certain percentage, up to a limit:

```
increaseBudgetBy: aPercentage upTo: aLimit
    budget := budget + (budget * (aPercentage / 100)).
    (budget > aLimit) ifTrue: [budget := aLimit].
    ^ budget
```

The *increaseBudgetBy:upTo:* message takes two arguments, represented by variables *aPercentage* and *aLimit*. This method changes the *budget* instance variable of the *Department* object that receives the *increaseBudgetBy:upTo:* message. In the second line of the method, you see a keyword message *ifTrue:* that functions as a conditional control construct. The receiver of this message is a Boolean value. The argument of the message is a block. A block is a sequence of one or more OPAL statements within brackets and is a first-class object in GemStone. The effect of a message *aBoolean ifTrue: aBlock* is to perform the code in *aBlock* if *aBoolean* is true. The third line of the method returns the new value of *budget*.

OPAL includes messages for iterative control structures, using blocks with arguments. Block arguments are

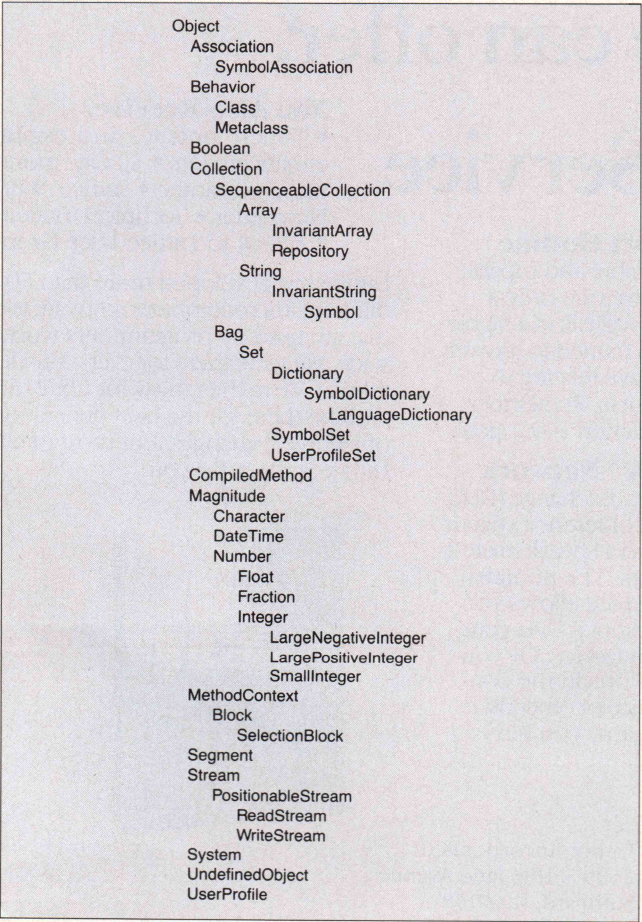


Figure 2: Kernel classes in the superclass hierarchy

declared at the beginning of a block. For example:

```
[n i (2 * n) - 1]
```

is a block with one argument, *n*. For this block to be evaluated, it needs a value for the argument. The *value:* message provides the argument and causes execution. A block returns the value of its last expression when executed. Thus:

```
[n i (2 * n) - 1] value: 7
```

returns 13. In general, given a value of *N*, this block returns the *N*th odd number.

All methods you have seen so far have been defined in terms of other messages. The definitions of methods are not completely circular. At the bottom of everything are primitive methods. When the OPAL interpreter encounters a message that has a primitive method, it executes a piece of machine code rather than an OPAL method. Primitive methods exist for arithmetic, comparisons, object creation and copying, array selection, string manipulation, and set functions. The set of primitive methods in GemStone cannot be augmented by a GemStone programmer, although such an extension could be provided.

Classes

Every class is represented by a class-defining object that describes the structure and behavior of instances of the class as well as the position of the class in the class hierarchy. Any object will return the CDO for its class in response to the message *class*. Suppose the variable *assoc* holds an object of class *Association*. (*Association* is

Classes		
PersonName	Ray Ross	
TitledName	Dr. Ray Ross	
TitledNameWithLetters	Dr. Ray Ross, OBE	

Class	InstanceVariables	Messages
PersonName	first last	first: last: fullName
TitledName	(above)+ title	(above) + titledName
TitledNameWithLetters	(above)+ letters	(above) + titledName (new Method)

Table 2: Representation of people's names with titles



QNX vs. OS/2 UNIX

QNX: Bend it, shape it, any way you want it.

ARCHITECTURE If the micro world were not so varied, QNX would not be so successful. After all, it is the operating system which enhances or limits the potential capabilities of applications. QNX owes its success (over 30,000 systems sold since 1982) to the tremendous power and flexibility provided by its modular architecture.

Based on message-passing, QNX is radically more innovative than UNIX or OS/2. Written by a small team of dedicated designers, it provides a fully integrated multi-user, multi-tasking, networked operating system in a lean 148K. By comparison, both OS/2 and UNIX, written by many hands, are huge and cumbersome. Both are examples of a monolithic operating system design fashionable over 20 years ago.

MULTI-USER OS/2 is multi-tasking but NOT multi-user. For OS/2, this inherent deficiency is a serious handicap for ter-

minal and remote access. QNX is both multi-tasking AND multi-user, allowing up to 16 terminals and modems to connect to any computer.

INTEGRATED NETWORKING Neither UNIX nor OS/2 can provide integrated networking. With truly distributed processing and resource sharing, QNX makes all resources (processors, disks, printers and modems anywhere on the network) available to any user. Systems may be single computers, or, by simply adding micros without changes to user software, they can grow to large transparent multi-processor environments. QNX is the main-frame you build micro by micro.

PC's, AT's and PS/2's OS/2 and UNIX severely restrict hardware that can be used: you must replace all your PC's with AT's. In contrast, QNX runs superbly on PC's and literally soars on AT's and PS/2's. You can

run your unmodified QNX applications on any mix of machines, either standalone or in a QNX local area network, in real mode on PC's or in protected mode on AT's. Only QNX lets you run multi-user/multi-tasking with networking on all classes of machines.

REAL TIME QNX real-time performance leaves both OS/2 and UNIX wallowing at the gate. In fact, QNX is in use at thousands of real-time sites, right now.

DOS SUPPORT QNX allows you to run PC-DOS applications as single-user tasks, for both PC's and AT's in real or protected mode. With OS/2, 128K of the DOS memory is consumed to enable this facility. Within QNX protected mode, a full 640K can be used for PC-DOS.

ANY WAY YOU WANT IT QNX has the power and flexibility you need. Call for details and a demo disk.

THE ONLY MULTI-USER, MULTI-TASKING, NETWORKING, REAL-TIME OPERATING SYSTEM FOR THE IBM PC, AT, PS/2, THE HP VECTRA, AND COMPATIBLES.

Multi-User	10 (16) serial terminals per PC (AT).	C Compiler	Standard Kernighan and Ritchie.
Multi-Tasking	40 (64) tasks per PC (AT).	Flexibility	Single PC, networked PC's, single PC with terminals, networked PC's with terminals. No central servers. Full sharing of disks, devices and CPU's.
Networking	2.5 Megabit token passing. 255 PC's and/or AT's per network. 10,000 tasks per network. Thousands of users per network.	PC-DOS	PC-DOS runs as a QNX task.
Real Time	2,800 task switches/sec (AT).	Cost	From US \$450. Runtime pricing available.
Message Passing	Fast intertask communication between tasks on any machine.		

For further information or a free demonstration diskette, please telephone (613) 591-0931.

Quantum Software Systems Ltd. • Kanata South Business Park • 175 Terrence Matthews Crescent • Kanata, Ontario, Canada • K2M 1W8

UNIX is a registered trademark of AT & T Bell Labs, IBM, PC, AT, XT and PS/2. PC-DOS and OS/2 are trademarks of International Business Machines. HP and Vectra are registered trademarks of Hewlett-Packard Company.

CIRCLE NO. 112 ON READER SERVICE CARD

OBJECT-ORIENTED PROGRAMMING

(continued from page 24)

a key-value pair used in building dictionaries.) The assignment:

```
ac := assoc class
```

causes *ac* to be assigned the CDO for class *Association*. CDOs respond to messages just as all other objects do. For example, CDOs respond to the *name* message. The result of the expression *assoc class name* is *#Association*. (Symbols are indicated with a # as a prefix.)

The instance variable names and methods for instances of a class are stored in the CDO for the class. CDOs also store the methods that instances execute in response to various messages. When an object receives a message, it consults the CDO to find out how to execute that message.

OPAL provides a class hierarchy to exploit similarities in structure and behavior of entities. (The built-in hierarchy is shown in Figure 2, page 24.) A subclass inherits structure and behavior from its superclass. Structure is inherited in that all named instance variables in the superclass are also present in any subclass. Suppose you wanted objects that represented people's names with titles. You could create a subclass *TitledName* of *PersonName*, and instances of *TitledName* would automatically have instance variables *first* and *last*. You could add an instance variable, *title*, to *TitledName* to

hold the title. (See Table 2, page 24.)

A subclass inherits methods from its superclass. Thus, if *fullName* is a message for *PersonName*, instances of *TitledName* respond to that message by using the method from *PersonName*. The lookup process to determine which method corresponds to a message starts in an object's class. If the message is not defined in that class, the search proceeds in the class's superclass, the superclass of that class, and so forth. Thus, the implementation of *fullName* can be overridden in *TitledName* if desired.

A subclass can implement messages of its own. For *TitledName* you might want a message that returns a string containing the full name with title:

```
titledName
^title + ' ' + self fullName
```

A new feature in this example is *self*, which is a special variable whose value is always the receiver of the message.

A subclass can also reimplement an inherited message. Suppose you define a class *TitledNameWithLetters* as a subclass of *TitledName*. This subclass adds an instance variable *letters* that holds the letters after a person's name, as in 'Dr. Ray Ross, OBE'. You can reimplement the *titledName* message in *TitledNameWithLetters* to include the letters after the name.

In OPAL, unlike some object-oriented languages, instance variables may be constrained to a kind of a class.

PC-SH PCMACS PC-UTIL

The look and feel of Unix on a PC!

PC-SH - \$4500

- includes Korn Shell features
- command history
- command editing
- command aliasing
- for, while, case, if, elif, else
- built-in commands
- debugging capability
- shell variables
- Unix regular expressions

PCMACS - \$4500

- full feature screen editor
- file encryption capability
- multiple file editing
- window size definition
- one key editor functions
- temporary file buffering

PC-UTIL - \$4500

banner	bdiff	cal	cat	cb	cd
cflow	chmod	cmp	col	comm	cp
crypt	cut	date	dd	df	diff
dirname	du	echo	env	expr	false
fgrep	file	find	grep	head	join
label	line	lpr	ls	make	mkdir
more	mv	mvdrr	od	pack	paste
pg	pr	proof	pwd	rm	rmdir
set	sleep	sort	split	strings	sum
tabify	tail	tee	test	time	true
touch	tr	tsort	uniq	unpack	units
unset	wc	whence	and more!		

SPECIAL OFFER — All 3 for only \$9900!

VEGAcon Corporation

P.O. Box 415 • Convent Station, NJ 07961-0415

Mail orders add \$1.50 P&H. N.J. residents add 6% sales tax.

VISA/MC orders: (201) 729-1696 — 30 Day Warranty

And, of course, full documentation is included!

*Unix and Ksh are trademarks of AT&T Bell Laboratories

*PC-SH, PCMACS, and PC-UTIL trademarks of Vegacon Corporation

CIRCLE NO. 113 ON READER SERVICE CARD

C programmers are talking about C_talk™

The easy way to add the POWER of OBJECT-ORIENTED Programming to C

C_talk extends your C compiler to a real Object-Oriented Language (OOL). It is not a new language; it simply adds Smalltalk-like features to C:

- ☐ Encapsulation
- ☐ Messaging (Dynamic Binding)
- ☐ Inheritance

C_talk offers all of the advantages of OOLs:

- ☐ A highly modular software design methodology
- ☐ Reusable software components
- ☐ Extendable software components

Plus the advantages of C:

- ☐ Speed, size, flexibility
- ☐ Ease of application delivery
- ☐ Access to C libraries and C tool sets

C_talk consists of an application development environment with:

- ☐ A powerful Smalltalk-like Browser for browsing, defining and editing an application's object class hierarchy
- ☐ A Preprocessor for converting object class descriptions into standard C programs that are compatible with popular C compilers
- ☐ An integrated, semiautomatic Make utility for controlling the preprocessing, compiling and linking of an application, object classes, C files or libraries

C_talk is designed to run on an IBM® PC (or compatible) with one of the following C compilers: Microsoft® C, Lattice C, Turbo C, or C86. A system configured with a hard drive and mouse is highly recommended.

To order:

CNS, Inc.

Software Products Dept.

7090 Shady Oak Road

Eden Prairie, MN 55344

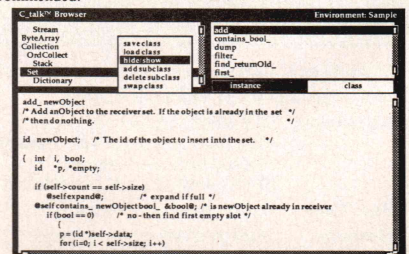
(612) 944-0170

Credit Cards: Master Card, Visa

Shipping: \$5 - US

\$25 - International

IBM is a registered trademark of IBM Corp.
MICROSOFT is a registered trademark of MICROSOFT CORP.
C_talk is a trademark of CNS, Inc.



CIRCLE NO. 114 ON READER SERVICE CARD

An object is a kind of class *foo* if its class is *foo* or a subclass of *foo*. If instance variable *personName* in class *Employee* is constrained to *PersonName*, then in instances of *Employee*, *personName* could be an instance of *PersonName*, *TitledName*, or *TitledNameWithLetters*. Constraints may also be placed on the elements of collection classes such as *Bag* and *Set*.

The Read-Write Boundary

In the same manner as conventional programming methodologies are not appropriate for today's applications, conventional data management systems are not appropriate for managing their data. Conventional data management systems are best at two things. One, they are good at batch processing large amounts of homogeneous data such as monthly credit-card billings. Two, they are good for high-transaction-rate applications such as ATM networks. The processing of data in applications such as CASE, cartography, and CIM fits neither of these two categories. The data in a cartographic database is not homogeneous, and transactions in a CASE system can take hours, or days, not just seconds.

Perhaps the best argument for integrating a data management system with an object-oriented language can be made by looking at the impedance mismatch across the interface between conventional application and database languages. Conventional application and database languages are incompatible both computationally and structurally.

Data manipulation languages use a read-write model

of interaction with application code. Database calls are embedded in the application language and allow for fetching, inserting, deleting, and modifying records or tuples. Although query languages have matured, allowing for the definition of complex views through sophisticated queries, they are by no means complete, and views are difficult to update.

In any event, the communication between application and database is declarative, not procedural. The application tells the database what it wants, not how to get it. This works well when what the application wants can be specified declaratively and the database can figure out how to get it in an efficient manner. All too often, though, an application needs to intersperse several database calls amongst application code to perform what should be a single logical operation.

A similar problem arises with regard to maintaining the consistency of a database. Many database systems do not allow constraints any more sophisticated than requiring key values to be unique. The application program ends up with the responsibility of maintaining all other constraints on the data.

Consider an office management system in which several applications reserve meeting rooms. In such a system, each application would first check for the availability of the room. This might involve fetching records to check that the individual reserving the room is authorized to do so and that the room has not already been reserved. The application would then post a record to indicate the reservation and perhaps post other

Playing by your own rules

Breaking the rules is a compromise. In software development breaking the rules can mean using non-standard "features" of a compiler, trying to fit your algorithm into a language that is not flexible enough. When you do break the rules, you must spend time justifying and documenting the change, and more time later correcting problems caused by that compromise.

We're Oregon Software and we propose an alternative: choose the rules you want. With Oregon Software's C++ compiler, you can select Kernighan & Ritchie C, ANSI C, or C++. You can use code you have already written in software projects that will extend into the 1990's and beyond.

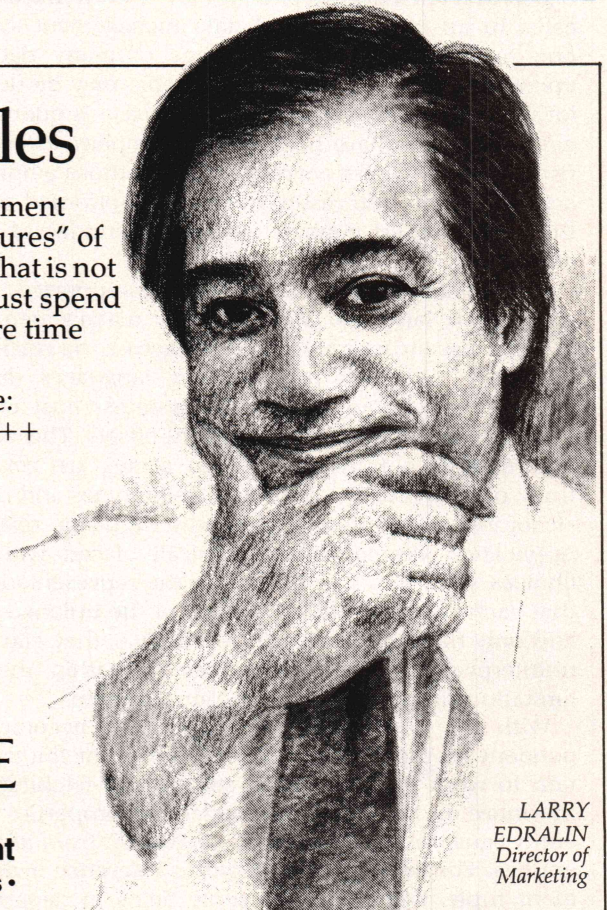
Breaking the rules is cheating – choosing your own rules is good planning.

To learn how to write your own rules, call us at 1-800-874-8501; or write Oregon Software, 6915 SW Macadam, Portland OR 97219.

OREGON SOFTWARE

Professional Products for Software Development

• Pascal-2 • Cross-Development • Modula-2 • C++ • SourceTools •



LARRY
EDRALIN
Director of
Marketing

CIRCLE NO. 115 ON READER SERVICE CARD

records to appointment calendars. A change to the structure of the database, or the policy for reserving rooms, might require locating and modifying every application that modifies the database. In an object-oriented data management system, a message could be defined that performed all necessary checks and updates to the database. In the event that the policy for reserving a room changes, the method for reserving a room could be easily located and modified.

You might argue that in a conventional system the code for reserving a room could be factored into a single procedure—thus, achieving the same effect. In a conventional system, however, none of the applications is forced to use the procedure when making a reservation. The encapsulation provided by object-oriented data management allows the designer of a class to control access to data in instances of the class. By organizing the methods for a class along with the structural description of its instances, the impact of changes in structure are localized. This control and organization becomes more important as the size and complexity of applications grow: placing the data in one file and the code that determines its meaning into several other files no longer makes sense.

Two other negative aspects of the read-write model should be pointed out. One, a procedure may need to make numerous calls to the database. If a procedure needs to fetch 1,000 records, it may need to make 1,000 calls. In an object-oriented data management system, one message can take the place of many database operations. Two, conventional systems may be doing a lot of unnecessary copying. If a database is queried for employees who have been with a company for at least five years, the tuples corresponding to those employees are copied into the result. In an object-oriented system the result would contain only the identities of those employees.

The differences between data types supported by the application language and the data management language cause structural impedance across the read-write boundary. Although programming languages directly support arrays, most relational systems must encode them by adding a field to represent offsets. This encoding requires a translation when arrays are retrieved from, or stored to, the database. Union types and nested structures are difficult to represent using relations. Application developers are typically faced with two choices. One, they can normalize the representation so that each relation stores only one of the unioned types and only flat structures are stored. Two, they can store uninterpreted bit strings if they fit within the size limitations imposed by the relational system.

With the first choice, the application becomes dependent on the decomposition in order for the translation to work properly when storing and fetching. Furthermore, in the relational model, the properties of an entity must be sufficient to distinguish it from all other entities. For an employee tuple to reference a department tuple, there must be some fields in department

tuples that uniquely and immutably identify departments. Using department names to identify department tuples is fine until a department's name changes. Maintaining the validity of foreign keys, such as a department name stored in an employees tuple, is known as referential integrity. Making up unique department numbers might solve the problem; however, this adds an attribute that may not be present in the world being modeled and is a burden on application developers. With the second choice, the database isn't providing any more functionality than a good file management system.

Suffice it to say that conventional programming languages manipulate data types by representing their structure in one place and distributing their semantics throughout the application code, whereas data management systems, because of their limited repertoire, can only manage data structures. An object-oriented data management system combines object-oriented programming with an integrated data model to free programmers from the tedious and error-prone coding needed to accommodate a separate data management system that understands neither the execution model of the language nor the structures used by the language in representing its data types.

An Object-Oriented Data Management System

Arguing in favor of an object-oriented language that provides a common abstraction for data definition, data manipulation, and general computation is one thing; actually developing one is quite another. What follows is a discussion of object-oriented data management issues and how they are addressed by GemStone.

One of the guiding principles in GemStone's development was uniformity; all objects, be they temporaries or shared and disk-resident, should be accessed in a uniform manner. This principle helped address the issue of how objects persist. Does an application explicitly state that an object is to be persistent and must be explicitly deleted, or do objects exist as long as they are reachable from a special root collection of objects? Some folks argue that persistence through reachability is not practical for a disk-based system: garbage collection of disk-based objects incurs too much overhead.

Let's take a look at the implications of explicit deletion. In the first place, who decides when an object can safely be deleted? In a shared environment, in which many applications are using the data, this is not an easy question to answer. When an object is deleted, what is to be done with all the references to that object? The problem is extremely similar to that of referential integrity in relational databases.

Assume that it would be satisfactory to replace these references with references to a special object *Nil*. Either all the references are located and replaced in one operation, which corresponds to garbage collection, or all references are screened for deleted objects, which corresponds to incremental garbage collection. If references are screened, are all the instance variables of an object screened when the object is accessed, or are only those instance variables to which messages are sent screened? The former involves greater overhead, whereas the latter allows the propagation of references to a

Some of the world's biggest problems are being solved with a touch of Smalltalk.

The French Ministry of Foreign Affairs is responsible for keeping

Abroad and involved in foreign affairs.

track of every French citizen living abroad and every foreigner living in France. Each day, they process thousands of requests for documents or information, each one of which takes at least fifteen minutes. Arthur Andersen, the world's largest accounting firm, has developed a natural language processing application with Smalltalk/V that enables clerks without computer training to extract the necessary data much faster. Thanks to Smalltalk and system developers Bart Schutte and Pascal Wattiaux, what once took fifteen minutes now takes 30 seconds. Vive la Smalltalk!

On the ground floor of high-tech environmental control.

Climate, energy, fire and security are all critical aspects of environmental control in large office buildings. The challenge for Johnson Controls, a leader in this industry, is to provide a control system that is both technologically advanced and simple to operate. Using Smalltalk/V, Research Scientists Gene Korienek and Tom Wrensch have created a workspace environment that allows rapid prototyping and modeling of future systems. At Johnson Controls this system is used to explore relationships between cognitive models of building operators and corresponding iconic representations of building components. Each system can then be tested by simply clicking a mouse and viewing the results in sophisticated color graphics on a PC.

The world is made of objects. So naturally, the world is turning to Object-Oriented Programming (OOPS). And the fastest, easiest OOPS language and environment is Smalltalk/V.

With OOPS you program by defining objects, their inter-relationships and their behavior. Objects can represent both real-world entities — people, places, things — as well as useful abstractions such as stacks, sets and rectangles. Smalltalk/V provides everything you need to solve problems big and small, including a comprehensive tutorial to get you started.

Who needs Smalltalk?

Because Smalltalk models the way people really think, it is perfect for scientists, engineers and professionals who have to solve tough problems in a

short amount of time. Perfect for programmers who are looking for a fast, efficient prototyping environment. And anyone who wants to quickly and easily learn OOPS.

Introducing Smalltalk/V286.

Our newest version of Smalltalk offers faster and more powerful OOPS capabilities. We've gone from 16 to 32-bit architecture. From 640K to 16 MB capacity for 25 times the memory. And designed it to run

on the next generation OS/2 operating system as well as DOS.

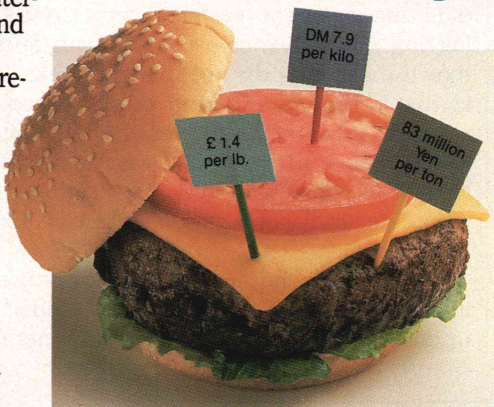
Get Smalltalk for a small price.

Smalltalk/V sells for just \$99.95. Smalltalk/V286 is \$199.95. The following optional applications packs are available for \$49.95 each: Communications; EGA/VGA Color; Goodies #1, Goodies #2, Carleton Tools and Goodies #3, Carleton Projects.

And everything comes with a 60-day, money-back guarantee.

So visit your nearest dealer. Or call toll-free, 800-922-8255 and order direct with MasterCard or Visa. Or write to Digitalk, Inc., 9841 Airport Blvd., Los Angeles, CA 90045.

And let us help you put Smalltalk into action.



Teaching students to think economically.

With Smalltalk, even non-programmers can create exciting applications. Economics Professor Arnold Katz of the University of Pittsburgh developed Economics PC Discovery World, an intelligent tutoring system for beginning microeconomics students. Using a mouse to access windows and manipulate data, a student can call up a set of markets and commodities for an imaginary community. By changing the scenario, the student can not only study a variety of market behaviors, but also test the validity of his or her own reasoning. A process that provides a lot of food for thought.

Smalltalk/V

digitalk inc.



deleted object: an object can be assigned to an instance variable without sending messages.

Given that explicit deletion does not incur less computational overhead, is difficult to control in a shared environment, and is less transparent to applications, persistence through reachability was chosen for GemStone. Instances of class *UserProfile* are used to represent properties of each user, including a list of dictionaries to be used in resolving symbols when compiling OPAL code for that user. Any object that is reachable from a dictionary in any user's *UserProfile* is persistent. Dictionaries are also used to manage name spaces and sharing. When an identifier is encountered and it does not correspond to a temporary, instance, or class variable, the dictionaries are searched to find an object corresponding to that identifier. A user may have any number of dictionaries to accommodate various degrees of sharing.

Data Integrity

Three kinds of failure can compromise the integrity of a database: an application may fail to complete because of a run-time error; the processor managing secondary storage may fail; and the media used to store objects may fail. Protection against media failure is achieved by replicating objects on disk.

Failures of the first two kinds require the careful posting of an application's changes to the database. By ensuring that either all, or none, of an application's changes are posted, inconsistent states of the database are avoided. This all-or-nothing form of posting changes is known as atomicity. Applications generally use *commit* and *abort* commands to define the changes that are to be atomically posted. When an application commits, an attempt is made to post all changes made by the application since the last commit or abort (you pretend that the first thing an application does is an abort). If all changes are successfully posted, then the commit succeeds; else the commit fails and the application is so informed. When an application aborts, all the changes made since the last abort or successful commit are thrown away.

The activity that occurs between an abort or commit and the succeeding abort or commit is known as a transaction. When a failure of the first two kinds occurs, it is treated as though the transaction had aborted.

When the processor fails, a lot of fancy footwork is required to guarantee that all the needed information is safely on disk, but it works.

The two basic choices for implementing atomicity are logging and shadowing. With logging changes are posted directly to the database and logged. When a transaction aborts, the log is used to restore the database to its previous state. A successful commit allows the log to be thrown away and a new log started. With shadowing, changes to an object are posted against a copy (a shadow) of the object. When a transaction aborts, the shadow copies of objects are thrown away. The database is left alone, as it has not been modified. For a transaction to successfully commit, all modified objects must be carefully replaced with their shadow copy.

Concurrency Control

Any system that allows concurrent access to shared data must handle conflicting changes made by applications running concurrently. Either conflicting changes are prevented (pessimistic concurrency control), or conflicting changes are discarded (optimistic concurrency control). Pessimistic concurrency control requires that a transaction state its intention prior to accessing an object by acquiring either a read-lock or a write-lock. While a transaction holds a lock on an object, other transactions are prohibited from acquiring a lock that would allow conflict. Optimistic concurrency control allows a transaction to proceed as though it were running alone. At transaction commit, the changes made by the transaction are checked for conflict with other transactions. If a conflict is detected, the changes made by the transaction must be undone.

As you may have noticed, concurrency control and data integrity are strongly related. Pessimistic concurrency control is generally bundled with logging, optimistic concurrency control with shadowing. Conventional data management systems tend to use locking and logging. The advantage of optimism and shadowing is that transactions do not manage locks. The down side is that arbitrary amounts of work can be lost when conflict is detected at commit. The situation is reversed for locking and logging. One further disadvantage of locking is deadlock: two transactions are unable to complete until each gets a hold of a lock the other is holding.

For the initial implementation, optimism and shadowing were chosen. This choice was guided by a desire for uniformity. As GemStone supports general programming

Employee	Person Name	String	String	Street Address	String	String
----------	-------------	--------	--------	----------------	--------	--------

Figure 3: Clustering of an Employee object

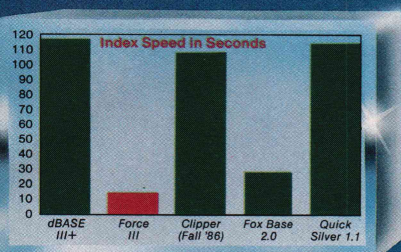
Employee	Person Name	String	String	Employee	Person Name	String	String	...
----------	-------------	--------	--------	----------	-------------	--------	--------	-----

Figure 4: Clustering of Employee objects omitting their addresses

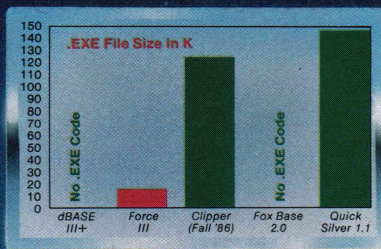
FORCE-III

The Ultimate dBase Compiler

FORCE III is the world's fastest dBase compiler. It compiles and links the quickest, and the .EXE files it creates run so fast you'll swear you wrote your program in C. To give you an idea how fast, take a look at the *Index Speed* graph below. This graph compares indexing speeds of the different products using a database which contained 8,000 records. The database was indexed on the numeric field, NUM.



Best of all, FORCE III creates very small .EXE files. The .EXE files are *true* native code files and do not rely on any kind of memory resident library or interpreter! The reasons the .EXE files are so small are



simple: FORCE III is an "intelligent" compiler and only uses those routines from the library that your program requires, and secondly, the entire library is written in Assembler, which makes the routines very small and very fast.

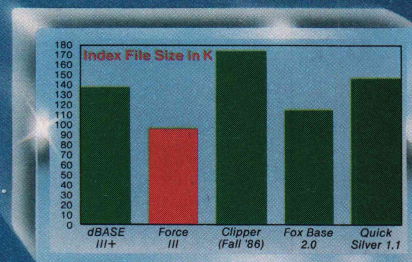
In the *Index Size* graph (below), the index files created by each product are compared. As you can see, not only is FORCE III the fastest at indexing, it also creates the smallest index files!

(Tests conducted on an IBM 8Mhz. AT with 20Mb hard disk.)

\$129

Royalty Free and fully linkable!

In addition to being the fastest true dBase compiler on the market, FORCE III generated .EXE files carry no royalties whatsoever! And FORCE III compiled programs can be linked to Borland's Turbo C and other C compilers using the Intel/Microsoft OBJ format. And linking to Assembler is a snap. We even included an ASCII graphics generator for easy screen and menu generation.



Arrays, UDFs, sound, new functions, and more!

Force III includes features you've come to expect in more expensive dBase compilers. For the programmer we have included I/O re-direction, FOR-NEXT loops, C like file primitives, and new data types. We've even included some Clipper compatible commands such as VALID.

Call now, 800-922-3001.



FORCE III comes with extras, like a 120 day warranty, an excellent reference manual and for the first 10,000 customers, we'll include the premier sourcework, **Advance dBASE III Plus Programming and Techniques** by Miriam Liskin (a \$21.95 value). All this for just \$129.00 (add \$5.00 s&h). In Colorado call 303-444-1542. In Europe call London at 44-1-631-0548. Or write: Sophco, Inc., PO Box 7430, Boulder, Colorado, 80306-7430.

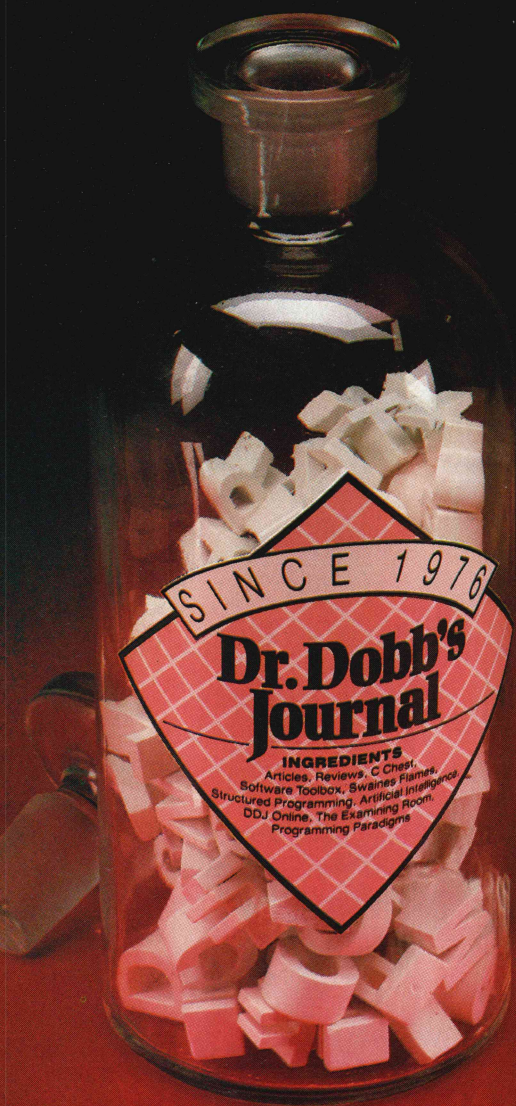
Products mentioned above are trademarked by their respective manufacturers.

THE CURE FOR COMMON CODE

Are you getting the recommended monthly allowance of C, Assembly, Forth, Pascal, BASIC or Modula-2? Subscribe to *Dr. Dobb's Journal of Software Tools* and you won't catch any nasty bugs again!

Each month the Doctor brings you aid for ailing algorithms and the cure for common code. For the latest developments in software design and pages of code that will make you a more productive programmer, take the Dr. Dobb's prescription.

For more than a decade, the programming elite have known *Dr. Dobb's Journal* to be the foremost source of software tools. Subscribe now and get your monthly dose from the Doctor.



PROLOG FOR
PASCAL
C BASIC
ASSEMBLY
MODULA-2

Dr. Dobb's Journal of Software Tools

The
R_x
for
Programmers

Subscribe
Now &

Save
Over
15%

Off the
Newsstand
Price!

SUBSCRIBE AND SAVE!
Subscribe to

DR. DOBB'S JOURNAL OF SOFTWARE TOOLS
and save over \$5—a 15% savings off the cover price!

☐ 1 year \$29.97 ☐ 2 years \$56.97

☐ Please charge my:

☐ Visa

☐ Master Card

☐ American Express

☐ Payment enclosed

☐ Bill me later

Card # _____ Exp. date _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

ONLY \$29.97! YOU SAVE OVER \$5.00

*Savings based on the full one-year newsstand rate of \$35.40. All foreign countries please add \$11 per year for surface mail; Canada & Mexico add \$28 per year for airmail; other countries add \$32 per year for airmail. All foreign subscriptions must be paid in U.S. funds drawn on a U.S. bank. Please allow 6-8 weeks for delivery.

A Publication of M & T Publishing, Inc.

362S

**\$5
SAVINGS**

SUBSCRIBE AND SAVE!
Subscribe to

DR. DOBB'S JOURNAL OF SOFTWARE TOOLS
and save over \$5—a 15% savings off the cover price!

☐ 1 year \$29.97 ☐ 2 years \$56.97

☐ Please charge my:

☐ Visa

☐ Master Card

☐ American Express

☐ Payment enclosed

☐ Bill me later

Card # _____ Exp. date _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

ONLY \$29.97! YOU SAVE OVER \$5.00

*Savings based on the full one-year newsstand rate of \$35.40. All foreign countries please add \$11 per year for surface mail; Canada & Mexico add \$28 per year for airmail; other countries add \$32 per year for airmail. All foreign subscriptions must be paid in U.S. funds drawn on a U.S. bank. Please allow 6-8 weeks for delivery.

A Publication of M & T Publishing, Inc.

362S

**\$5
SAVINGS**

COMMENTS & SUGGESTIONS

Dear Reader,

March 1988, #137

Dr. Dobb's has a long tradition of listening to its readers. We like to hear when something really helps or, for that matter, bothers you. In this hectic world of ours, however, it is often difficult to take time to write a letter. This card provides you with an easy way to correspond and, if you include your name and address, we may use appropriate comments in The Letters column. Simply fill it out and drop it in the mail.

—Ed

Which articles or departments did you enjoy the most this month? Why?

Comments or suggestions _____

Name: _____

Address: _____



BUSINESS REPLY MAIL

FIRST CLASS PERMIT 790 REDWOOD CITY, CA

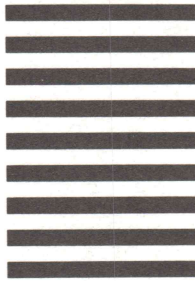
POSTAGE WILL BE PAID BY ADDRESSEE

Dr. Dobb's Journal of
Software Tools

Box 3713
Escondido, CA 92025-9843



No Postage
Necessary
If Mailed
In The
United States



**Dr.
Dobb's
Journal
of
Software
Tools**



BUSINESS REPLY MAIL

FIRST CLASS PERMIT 790 REDWOOD CITY, CA

POSTAGE WILL BE PAID BY ADDRESSEE

Dr. Dobb's Journal of
Software Tools

Box 3713
Escondido, CA 92025-9843



No Postage
Necessary
If Mailed
In The
United States



**The
R_x
for
Programmers**

**Subscribe
Now &**

**Save
Over
15%**

**Off the
Newsstand
Price!**

PLACE
STAMP
HERE

Dr. Dobb's Journal of
Software Tools

501 Galveston Drive
Redwood City, CA 94063

and data manipulation in a single language, the objects in a GemStone database span the continuum between objects that correspond to values of variables in conventional programming languages and those that correspond to large design objects as may be found in VLSI databases. Conflict is unlikely on objects that correspond to program variables. For one thing, they are unlikely to be shared or persistent. Imposing pessimistic concurrency control on objects at this end of the continuum is an unnecessary burden.

Optimism and shadowing allow GemStone to provide each transaction with a private workspace, within which the activities of other transactions cannot affect the transaction's progress. At commit, privacy is abandoned, changes in the private workspace are made available to other transactions, and changes committed by other transactions become visible. Servio Logic Corp. realized, however, that in the case of large design objects, the amount of work that must be undone when conflict is detected may be unacceptable. They therefore began investigating supporting pessimism control in addition to optimism.² The goal was to make pessimism optional and, to as large a degree as possible, not of concern to applications that chose not to use it. Support for pessimism will be included in a forthcoming release.

Concurrency control is one of the areas in which object-oriented data management offers great promise. Recent research efforts in programming languages have explored the notion of behavior-based concurrency control. As a simple example, consider a savings account to which deposits are credited and withdrawals are debited. There is no intrinsic reason why two transactions that are both crediting a given account need conflict. Neither transaction is concerned with the final balance after the credit. All they care about is that the proper amount is credited to the account; the order in which the amounts are credited is not of concern. Basically, each transaction adds the amount deposited to a list of credits to the account. The credits are processed in the order in which they were added to the list. That the credits were added to the list in the opposite order to that in which the two transactions committed has no effect upon the desired behavior. The read-write barrier of conventional systems prevents this form of concurrency as the database has no knowledge of the semantics of operations beyond read and write.

Large Objects and Large-Object Space

Although relational systems can support a large number of tuples, they generally do not allow tuples to be larger than a page. Object-oriented systems must support large objects as well as a large numbers of objects. If large objects are not supported, application developers will have to encode large objects into objects no larger than those supported by the system. A GemStone system supports 2^{31} objects (2^{32} counting instances of *SmallInteger*), and an object can contain 2^{31} instance variables.

When an object is, or grows, larger than a page, it is broken into pieces and is no longer stored contiguously.

Large objects can be accessed and updated without bringing the entire objects into memory. Large objects can also grow and shrink without copying the entire object.

The basic data formats provided by an object-oriented system must support reasonably direct and efficient implementations of user-defined classes. Although relational systems support both records (tuples) and sets (relations), arrays must be encoded. A data management system must support all three.

The underlying basic storage formats must in turn efficiently support the basic data formats. GemStone supports five basic storage formats—self-identifying (for example, *SmallInteger*, *Character*, byte (for example, *String*), named, indexed, and nonsequenceable collections. The byte format is used for classes whose instances may be considered unstructured. Structure is imposed by the methods that operate on the objects. The indexed format supports access to components of an object by integers, as in instances of *Array*. The byte and indexed formats support, without copying changes in the size of an object. The named format supports access by instance variable names. Classes whose instances have both named and indexed instance variables are supported by a hybrid format. The nonsequenceable collection (NSC) format supports collection classes such as *Bag* and *Set*. The members of such collections are not identified by name or index; instead, collections can have members added, removed, or enumerated, and efficiently support union, intersection, difference and tests for membership.

Two other areas that need to be addressed by data management systems are clustering and associative access. Clustering is the placement of objects that tend to be accessed together near each other on the disk. The objects are placed on as few, preferably contiguous, pages as possible. By so placing the objects on disk, fewer disk accesses are required to bring these objects into memory.

Consider the *Employee* object of Figure 1. If the *clusterDepthFirst* message were sent to the object, its layout on disk would be as in Figure 3, page 30. Since the *ssNo*, *stNumber* and *Salary* instance variables are selfidentifying, they are directly represented in their containing objects and need not be clustered. Now consider a collection of *Employee* objects. If enumeration of the elements of the collection occurs frequently, the elements can be clustered together. If these enumerations tend not to access the addresses of employees, it may be desirable to omit addresses from the clustering. The layout on disk of the employees would be as in Figure 4, page 30. By omitting addresses from the clustering, employees and their frequently accessed instance variables can be enumerated with even fewer disk accesses. OPAL's clustering protocol is flexible enough to allow such clustering.

An associative access is a search of a collection based upon the internal state of the collection's elements—for example, a collection of employees can be associatively accessed for employees who live on a certain street. Even with clustering, searching large collections by a sequential scan may yield unacceptable performance.

Associative accesses should take time that is no more than logarithmic on the size of the collection. Hashing would allow the associative access of an employee with a particular social security number in nearly constant time, regardless of the size of the collection being searched. B-tree indexes support associative access in time that is logarithmic on the size of the collection and efficiently support range queries, such as locating those employees whose salary is within a given range.

There are many issues with regard to associative access in object-oriented systems.³ For example, should indexes index all instances of a class or only instances of explicit collections? How is the use of indexes to be indicated? Should indexes be based upon the structure or protocol of objects? If by structure, should indexes be on identity or value? GemStone supports B-tree indexes into explicit collections using the structure of objects. Both identity- and value-based indexing is supported. A limited calculus sublanguage is provided to make use of indexes. The language was constructed so that associative queries may be viewed as procedural OPAL code.

State of the Technology

Object-oriented languages and data management are emerging technologies. The first commercially available data management systems have only recently arrived on the scene. Deciding when and if to jump on the bandwagon is difficult. Strong interest in object-oriented systems, as indicated by the Conference on Object-Oriented Systems, Languages and Applications (OOPSLA) becoming ACM's third largest conference in just two years, is not sufficient. Other promising technologies of the past have failed to yield their expected benefits.

If, however, you are encountering the frustration with structured programming I discussed in the introduction, you might just browse through the proceedings of the OOPSLA conferences^{4,5} to see what advances are being made and the kinds of applications being developed using object-oriented systems. You may well be surprised at the significant applications being developed with object-oriented technology. GemStone, for example, is being used by an agency of the U.S. government to develop a new, automated coastal chart production and maintenance system. In this system, GemStone manages both the static feature information and the procedural knowledge to render that information into readable and reliable charts. At another U.S. government facility, dedicated to research in CIM, GemStone is being used to manage information flow between the many, often incompatible, systems involved in computer-integrated manufacturing.

You may find that object-oriented technology is more mature than you thought. You might start asking hard questions about performance and how to compare object-oriented systems. Many argue that part of the price to be paid for the benefits of object-oriented systems is an apparent increase in the consumption of machine resources. It may well be the case, however,

that many applications can be developed using today's technology and still yield adequate performance. After all, an application that can be developed quickly, that is easily maintained, and whose performance is adequate is often preferable to missed production deadlines, buggy code, and blazing performance.

In the same manner as relational systems have markedly improved their performance over the last decade, so will object-oriented systems. Hardware will continue to get faster and cheaper. If performance isn't quite adequate on the hardware you are running today, it will be tomorrow.

Convinced? Want one? Which one? That may be a hard decision. For one thing, there is no single object model. In some models the collection of all instances of a class is meaningful; in others it isn't. Some models support explicit deletion; others don't. There are object-oriented extensions to C. Will these C extensions perform better than systems whose heritage is Smalltalk? If so, there may be a price, such as the slower development and more difficult maintenance implied by a compile-and-link methodology. Perhaps what you really need is the EXODUS extensible database system being developed at the University of Wisconsin or the POSTGRES system being developed by Stonebraker and crew at Berkeley.

What about benchmarks? A good set of benchmarks would make comparison shopping easier. There are problems too numerous to mention here. It's difficult enough developing benchmarks for relational systems or any given programming language. It's even harder for systems that integrate different language extensions with different object models.

See you at OOPSLA.

This article was condensed by the author from Development and Implementation of an Object-Oriented DBMS by David Maier and Jacob Stein.

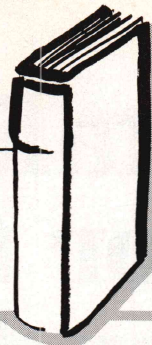
Notes

1. J. Diedrich and J. Milton, "Experimental Prototyping in Smalltalk," *IEEE Software*, vol. 4, no. 3 (May 1987).
2. D. Maier and J. Stein, "Indexing in an Object-Oriented DBMS," *Proc. International Workshop on Object-Oriented Database Systems* (Asilomar, Calif.: IEEE Computer Society Press, September 1986).
3. D. J. Penney, J. Stein, and D. Maier, "Is the Disk Half Full or Half Empty?: Combining Optimistic and Pessimistic Concurrency Mechanisms in a Shared, Persistent Object Base," *Proc. Workshop on Persistent Object Stores* (Appin, Scotland, August 1987).
4. *Proc. of ACM Object-Oriented Programming Systems, Languages and Applications Conference (OOPSLA-86)* (Portland, Oreg., October 1986.) Also published as *ACM SIGPLAN Notices* vol. 21, no. 11 (November 1986).
5. *Proc. of ACM Object-Oriented Programming Systems, Languages and Applications Conference (OOPSLA-87)* (Orlando, Fla., October 1987). Also published as *ACM SIGPLAN Notices*, vol. 22, no. 12 (December 1987).

DDJ

Vote for your favorite feature/article.
Circle Reader Service **No. 1.**

D ICTIONARY



A **Dictionary** is a collection of data pairs, much like an English language dictionary is a collection of term/definition pairs. The "term" entry is the key and the "definition" entry is the value of the pair, or association.

A Dictionary functions like a single-key database. This is how you create an object of class Dictionary with room for two key/value pairs:

```
Worker1 := new(Dictionary, 2);
```

Next fill the dictionary with strings as keys like "Name" and "Age" and the corresponding values, the string "Sam Jones" and the integer 22. You are free to mix data types.

```
Worker1["Name"] := "Sam Jones";  
Worker1["Age"] := 22;
```

Like a database, an Actor dictionary grows when you add to it more entries than you originally specified. Below, the dictionary grows from 2 to 4 as you add two more entries, or pairs:

```
Worker1["Department"] := #Engineering;  
Worker1["Reviews"] := #(85 73 98 94 100);
```

Mixing data types allows great flexibility. You can even incorporate the above dictionary, Worker1, as a value in an entry of another dictionary, as in Employees below:

```
Employees := new(Dictionary, 10);  
Employees["Sam"] := Worker1;
```

Access the data in a dictionary by specifying the appropriate keys. The following statement returns the array #(85 73 98 100):

```
AWorkersReview := Employees["Sam"]["Reviews"];
```

You can also retrieve data from dictionaries by enumerating over the dictionary and extracting particular entries. Here is how to create a separate "database" of employees working in the Engineering department:

```
Engineers := extract(Employees, {using(employee)  
employee["Department"] == #Engineering});
```

Another feature allows you to create a new database incorporating only a subset of the original's data. You do it by enumerating over a dictionary and collecting particular data about every entry:

```
MailingList := collect(Employees, {using(employee) employee["Address"]});
```

Dictionary objects are available in Actor as ready-made units of functionality. They can be modified or specialized for any application. However, as programmers, you are never involved in a dictionary's physical implementation or memory management.

Technical Specifications

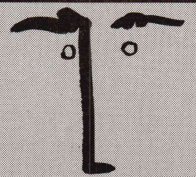
- Runs with Microsoft Windows 1.x, 2.0 and 386.
- Pure, single-inheritance object-oriented language, incrementally compiled.
- Dynamic linking to C, Pascal, Assembler, or Fortran libraries. Pass data in C structures.
- Pascal and C-like syntax.
- Programming tools: Browser, Inspector, Debugger, File Editor.
- Full access to MS-Windows systems calls, multi-tasking, and DDE.
- Fast device-independent graphics: lines, shapes, icons, cursors, bitmaps, metafiles, Turtle graphics, sample control language using YACC.
- 150 classes, 1500 functions, fully extensible.
- Window styles: tiled, overlapping, popup, child, edit, dialogs. Controls: list boxes, scroll bars, buttons, check boxes.
- Data structures: stacks, arrays, queues, lists, dictionaries, sets, sorting, hashing, intervals.

- AI support: frames, symbols, dictionaries, lists, symbolic programming, functional arguments. Parsing and lexical analysis YACC compatible.
- String manipulation: substring, concat, append, insert, remove, search.
- 643-page manual includes tutorial and reference.
- No license fees. Generates stand-alone applications.
- Fastest interactive OOL available.

Prices

Actor \$495 • Academic \$99 • Academic site license \$99 • Manuals for site license \$35 • **New!** Language Extension 1 \$99 • Shipping \$5 US, \$25 Int'l
Tech Support: Level 1 (20 phone calls) \$100 • Level 2 (Developer's package) \$250
Training Courses: 201 Intro. to Actor and OOP (2 days) \$395 • 202 Actor Developer's Workshop (3 days) \$795 • 203 Both 201 and 202 (1 week) \$995

ACTOR[®]



Class of the Month

Vol. 1

No. 1

Actor: an interactive object-oriented programming environment running under Microsoft Windows.

Class: a template for data structures, or objects, with similar properties. These objects are tested, refined and reusable units of functionality you can snap into your Actor programs at any time.

The Whitewater Group[®]
Technology Innovation Center
906 University Place
Evanston, Illinois 60201

For more information call:
(312) 491-2370

Actor is a registered trademark of The Whitewater Group, Inc.

An Italic Font in C for the EGA and VGA

A custom font for the EGA in Turbo C with source code for a resident program that makes the font permanent

by Andrew J. Chalk

Innovative screen displays sell software, which is one reason why so many developers provide demonstration disks, often free. One thing that makes a program distinctive is having a custom typeface for the text that it displays. Until the advent of the EGA, this was a luxury reserved for programs that operated in graphics mode.

Developing a nongraphics application to operate in graphics mode (for example, Framework in its EGA two-color graphics mode) is more costly than using the strong text mode support in the PC, however. Furthermore, prior to the EGA, graphics (with the exception of Hercules) were too poor to be very attractive. The EGA was such a rarity a year after its introduction that Peter Norton could write in his authoritative 1985 *Programmers's Guide to the IBM PC* that "we won't be discussing the 64-color palette of the EGA/ECD combo because it's quite rare and specialized and doesn't really fit into the mainstream of the PC family" (page 77).

This situation changed with the advent of EGA clones from Chips

and Technologies and others. At the present time, a no-frills EGA card sells for around \$150 and will probably cost around \$100 before the end of 1988. At least for a while, the EGA will be the de facto video standard, so it behooves developers to take advantage of its special features.

In this article I show how to exploit one of the interesting features of the EGA—the ability to replace the standard character font in text mode with one of your own choosing. As an example, I use an italic font and provide source code for a TSR that loads the font into RAM so that DOS and (most) application programs can use it. The TSR is necessary because the EGA reloads the ROM character set when the video mode is changed. By intercepting the BIOS for video mode changes, you can reload your custom font in its place. The TSR can be deinstalled, freeing up the 17K of memory it occupies for other programs and preventing incompatibilities.

The source code is written in C (Borland International's Turbo C, to be precise), so this article also serves a second purpose—explaining some of the techniques for writing resident code in high-level languages. As I will explain later, I have concluded that this is basically a bad

idea. If the on-line services and bulletin boards are good indicators, however, it is a subject of great interest, so it is worth spreading the techniques just so that others can become similarly disabused of the practice.

In order to use the code included here, you first need to understand fonts on the EGA. Then I'll talk about the C implementation in the source code. After that I'll discuss the TSR aspects of the source code and why you should write TSRs in assembly language.

Fonts on the EGA

Programmers writing for the monochrome display adapter (MDA) and the color graphics adapter (CGA) are stuck with the character sets provided in those board's ROMs. If you want a different character set, say as users of APL do, you have to replace the ROM. On the EGA, things are different. The fonts are "soft," meaning that although the ROM character generator is used by default, it can be replaced by a character set of your choosing. In fact, the EGA can support four character sets in what IBM calls four different blocks. Normally, you use block 0.

The EGA has BIOS support for the loading of an alternate character set

Andrew J. Chalk is president of Magna Carta Software, P.O. Box 475594, Garland, TX 75047.

through interrupt 10h, function 11h, subfunction 0. You make a call to this function with *ES:BP* pointing to a table containing your font (in a format I will explain), *DX* set to the ASCII ordinality of the first character in your character set, *CX* set to the number of characters in your character set (maximum 100h), *BH* set to the number of bytes per character, and *BL* set to the block to load (usually 0). These parameters provide the BIOS with sufficient information to load your font because of the way that fonts are stored.

Figure 1, below, shows a letter g as a magnified version of its screen image and as a stored character in memory. The figure assumes an enhanced color display in 25-line mode, in which case each character is 14 scan lines high and 8 pixels wide. A 25-line screen therefore fills 14×25, or 350 scan lines, as does the EGA. On the monochrome display, a 14×9-character box is used, and on the regular color display, the CGA 8×8-character box is used. The 43-line mode that is popular on the EGA driving an enhanced color or a monochrome display is achieved by loading the 8×8 ROM character set (because 8×43 is 344, just less than the 350 scan lines available). BIOS support exists for this, too, although two bugs in the original EGA BIOS make its implementation too big a subject to digress into here.

Let's assume that the EGA is driv-

ing an enhanced color display. In this case, each character is 14 scan lines high and 8 pixels wide. Representing this in RAM is simplified by the fact that each pixel that forms part of a character can be considered to be either "on" or "off" when a given character is on the screen. This means that the state of each pixel can be represented in binary by 1 bit. Furthermore, the designers of the EGA seem to have chosen a character width of 8 because this permits each character-scan line to be represented by exactly 1 byte.

In Figure 1, the hexadecimal values of each scan line are shown above the character. The arrows that lead in the direction of memory show that the letter g is stored as 14 contiguous bytes of data. Because g has an ASCII value of 67h, the characters next to it are the ASCII values 66h and 68h. The latter of these is h.

When you load a custom font into the EGA, you tell the BIOS, through *ES:BP*, the address of a buffer containing your chosen characters. If you want to load a complete character set, this buffer is 3,584 (14×256) bytes long. The EGA lets you load fewer than 256 characters, and it lets you choose the starting position in the ASCII sequence. The sequence of characters for any one load operation must be consecutive members of the ASCII character set, however, or you will get garbage on the screen.

Note two important limitations of the EGA font features. First, characters have a fixed width (but may vary from 1 to 32 scan lines high), so you do not have the same flexibility as in graphics modes. Second, although your 14×8 fonts work fine on a monochrome monitor in 25-line mode, a different character set is required if you support different numbers of screen lines. Suppose, for example, you were going to incorporate a feature into a database such that the user could press a hot key and immediately switch into 43-line mode and thereby see more records in "table view." If you had a custom typeface, you would need a 43-line-mode equivalent of it that would be loaded at the same time.

Notwithstanding these limitations, the custom font capability of the EGA is impressive. As I stated earlier, a video mode reset restores the ROM character set, so your application should perform a reload operation every time it performs a video mode reset. Furthermore, if you want to load your custom fonts only in certain video modes, you should check the video mode before loading. The example program *ITALIC.C* in Listing One, page 50, only loads the custom character set in modes 0-3 (text modes) and 7 (monochrome).

A Resident Italic Font

The example program consists of

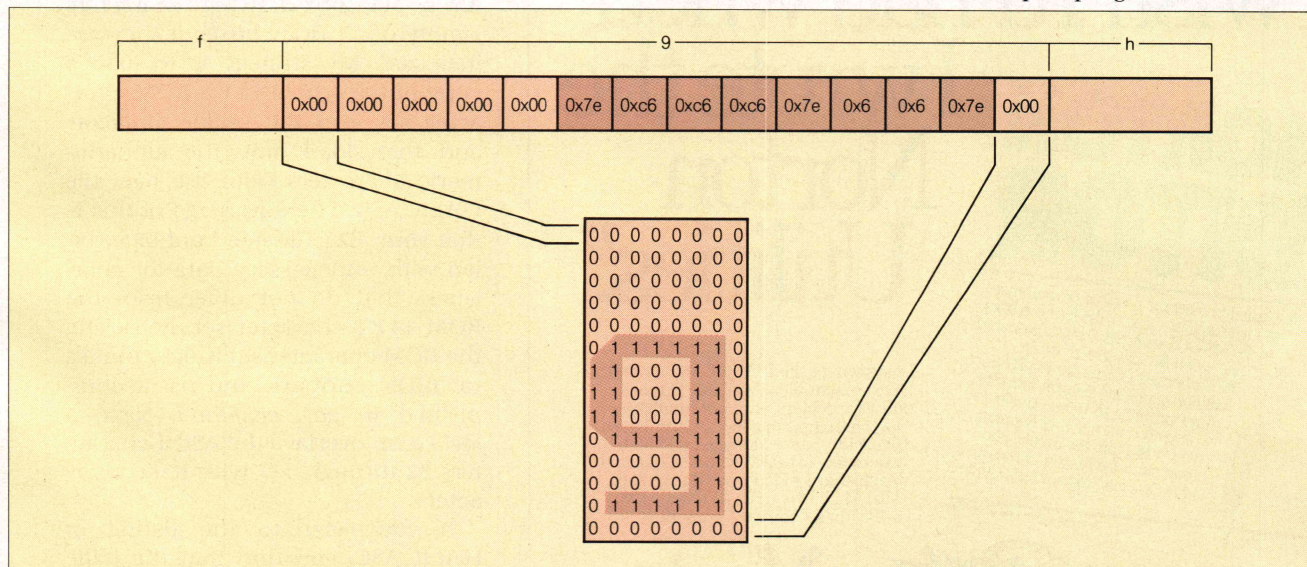


Figure 1: EGA character representation on the screen and in memory

ITALIC FONT IN C (continued from page 37)

two parts: ITALIC.C is the source code, and ITALIC.ASC (Listing Two, page 52) contains the code for the font in a form I will explain shortly.

The program first checks the system configuration to see if custom fonts are supported. This is done by means of the function `get_video_info()`, which first checks for the presence of an EGA in the system through the recommended BIOS call (function 12h). If an EGA is not present, the value of `BL` is returned unchanged, in which case you exit with a message explaining to the user that an EGA is required. There is a practice in the literature of checking for the IBM signature in the EGA BIOS. Not only is this kludgy but it is also specifically disapproved of in the EGA BIOS listing.

It is not enough to know that an EGA is present—it must also be active. The user may, for example, have an EGA driving a color monitor and an MDA driving a monochrome monitor and be using the MDA. To

see if the EGA is active, you check that bit 4 of the EGA information byte in the ROM data area at 0:0487h is 0. If not, you exit with a message to the user explaining that the EGA must be active.

If space had permitted, I would have included code to save the video configuration and switch adapters. The saved information could then be used to restore the *ex ante* state of the machine on exit. For the same reason, I have also omitted code to detect a switch of adapters while ITALIC is resident. Be warned: ITALIC illustrates a technique; it is not a full-blown professional program.

Having determined that an EGA is active, you then determine whether it drives a monochrome or a color monitor. The value of `BH` is 1 if monochrome and 0 if color. The result is used to set the global variable `ega_color` appropriately. If a color monitor is in use, you test for an enhanced color display. If it's not present, you exit with an explanatory message because the EGA will use the 8×8 font on a regular color display and my example requires 14

scan lines per character.

If the system checks out, you return to `main()` and set the video mode based on the type of monitor in use. You then check whether a copy of the program has already been installed. This involves scanning down through memory from the program segment prefix (PSP) for two identification words inserted in the global data area near the top of the program. These words are `our_id1 = FACH` and `our_id2 = 1000h`. The function `already_installed()` first peeks at the offset of `our_id1` with the segment value equal to 1 less than the PSP. The segment value is then decremented until 0 is reached or a match is found. If a match is found, the next word is peeked at and compared with `our_id2`. If you assume that all characters are equally likely, the chance of erroneously concluding that ITALIC is resident when it is not if you load halfway up memory on a 640K machine is 1 in 13,158. If you are uncomfortable with this, you can lengthen the odds by searching for more than two words.

If you find a copy of ITALIC, you deinstall it and print a message telling the user. The mechanics of deinstalling a resident program are explained in the next section. If ITALIC is not present in memory, you can proceed with installation. There are three distinct phases of this process.

First, you only want to replace the alphanumeric characters in the ASCII set. Box-drawing characters simply don't draw boxes if they are italicized. My strategy is to load a copy of the whole 14×8 ROM character set into the buffer `fontarray` and then load only the alphanumeric characters from the data file ITALIC.ASC. The advantage of this is that your .EXE file need not be swollen with unnecessary data for characters that do not differ from the ROM 14×8-character set. Retrieving the ROM character set is easy thanks to BIOS support and is accomplished in `get_egafont()`. Next, a `for()` loop overlays the ASCII characters 32 through 127 with italic characters.

If you refer to the listing of ITALIC.ASC, you find that the italic font data is set up as a two-dimensional array with each row consist-

"This is the programmer's editor I wish I'd had when I wrote the Norton Utilities."

THE NORTON EDITOR

- Created to meet the needs of programmers.
- Lightning fast—one of the fastest editors in the MS-DOS world.
- Easily customized and saved.
- Split-screen editing.
- Condensed/Outline display.
- Structured programming features, including Auto-indenting.
- Word features for writing documentation.
- Supports mouse.
- From the people who brought you the Norton Utilities and Norton On-Line Programmer's Guides.

For the complete IBM® PC family and compatibles.

Designed for the IBM® PC, PC-AT and DOS compatibles. Available at most software dealers, or direct from Peter Norton Computing, Inc., 2210 Wilshire Blvd., #186, Santa Monica, CA 90403. To order: 800-451-0303 Ext. 40 (Visa and MasterCard welcome), 213-453-2361. MCI Mail: PNCL. Fax 213-453-6398. ©1987 Peter Norton Computing.

Peter Norton

COMPUTING

CIRCLE NO. 119 ON READER SERVICE CARD

ing of the appropriate hexadecimal values for a single character. The listing shown is actually the output of FONTEDIT, a full-screen, real-time EGA font editor included in C Windows Toolkit (see the "Availability" section at the end of this article). Don't worry about typing the listing; download details are given at the end of the article.

The second stage is the loading of your font (telling the EGA to use it) using interrupt 10h, function 11h, as described earlier. This is accomplished by the function `load_user_egaxfont()`. At this point the screen display immediately changes to reflect the new font.

The third stage is the resident installation of a replacement interrupt handler for interrupt 10h so that you can detect video mode changes and reload your font if necessary. First, you save the PSP and environment pointer in the PSP at offset 2ch to use for later deinstallation. Next, you save the old interrupt 10h address using `getvect()` and install your own handler with `setvect()`. Finally, you terminate and stay resident (more on this in the next section).

The new font affects every character on the screen. The italic font presented here is probably not distinct enough for serious text work, but it could be edited to be so. The italic font in Microsoft Word is created through exactly the same kind of techniques. Other fonts along the lines of the large selection supplied with the Hercules Graphics Card Plus that are practical fonts for text-intensive work are also possible.

Programming Resident Programs in High-Level Languages

As I stated in the introduction, the experience of writing this (simple) TSR in C has lead me to conclude that such programs are best written in assembly language. There are several reasons for this. Perhaps the most important one is the sheer size. ITALIC occupies 13K RAM, of which 3,585 bytes are the font buffer, 1,344 bytes are the replacement font data, less than 100 bytes are for other global data, and 512 bytes are for the .EXE header. The remaining 11,500 odd bytes are "code." It is

this portion that assembly language could shrink down to perhaps 3,000 bytes (I have not done it for this program). A second reason in favor of assembly language is the irrelevance of the portability issue.

An argument often legitimately raised in favor of high-level lan-

It can hardly be argued that the problems arose from the choice of language.

guages is their advantage in terms of development time. In fact, for resident code, even given that I was working without the benefit of a large literature on the ins and outs

of programming TSRs in high-level languages such as exists for assembly language, so many problems resulted from having a compiler between me and the machine that I spent a lot of time inside the debugger. Assembly-language TSRs are easier to debug.

It can hardly be argued that the problems arose from the choice of language. Wasn't it the low-level links that made C so popular as an application language? It is also difficult to argue that the choice of compiler was the problem. Turbo C has library support for all the function calls associated with resident code. Although it lacks a built-in debugger at the time of writing, Periscope does an admirable job. In-line assembly language is also available, but the objective here was to not use a single line of in-line code (that isn't really writing a TSR in a high-level language).

It might be argued that programming TSRs obviates the need to learn assembly language. The last person I would recommend to write a TSR in a high-level language is someone who did not know assem-

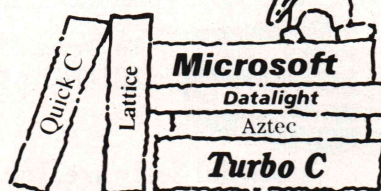
UNIX/C WINDOW DEVELOPMENT COMPATIBILITY with CURSES for MS-DOS and MS-OS/2.

THE BETTER PRODUCT. "Aspen Scientific's Curses library is a fine PC version of System V Curses. Screen updating was fast and clean... has good documentation and is available for many compilers...less expensive... its greater flexibility makes it an attractive package for developers." *Computer Language, June/87*
"This is a nice product. If you need Unix-compatible screen output in your programs, or if you just want a nice clean window-management package, I'd recommend it." *Allen Holub, Dr. Dobb's Journal, August/87*

Limited Time Offer:
CALL 1-800-255-5550
ext. 171
to **ORDER CURSES NOW**
and receive FAST Unix
compatible forms tool
kit with source code
FREE

NEW!
Window/Menu Manager Option

Complete curses tool kit: **\$119.**
Source code available for: \$289.



ASPEN SCIENTIFIC

P.O. BOX 72 WHEAT RIDGE,
COLORADO 80034-0072
For technical questions please call
(303) 423-8088

Power Graphics

Essential Graphics Takes You To New Heights Of Graphic Programming In C. Increases Speed 40%.

When first brainstorming this ad I spent a considerable amount of time trying to determine what graphic image to use as an illustration. The Space Shuttle, Mona Lisa, Robo-Cop - there are so many available.

Then it occurred to me. When you have the fastest, smallest functions, it's really irrelevant to show a complicated graphic image. It would be as if thinking up a sexy graphic were the test of a library.

The Graphics Test

The crucial test of a professional graphics package is: are the functions powerful, reliable, fast, and do they truly eliminate grunt work?

How quickly the functions execute is the criterion most people look for in a graphics library. There is no sense paying for a package that is not up to speed.

Beware Of Speed Traps

We eliminate the bios calls and write directly to the graphics card. As a matter of fact, in a recent benchmark, we were clocked 40% faster than our nearest competition.

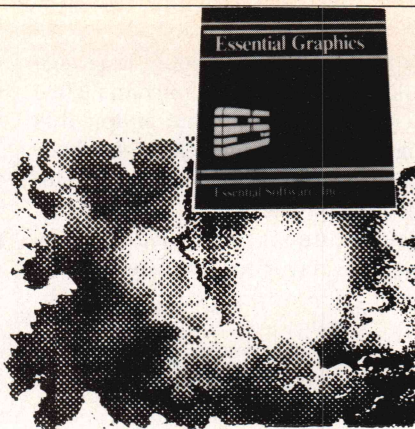
I'd like to repeat that "...clocked 40% faster than our NEAREST competition." Please take a moment to think about the significance of that speed increase in the project you are contemplating or working on now.

Our efficient, granular coding provides you with code sizes up to 75% smaller. Lean, fast and tight - just the way you would have done it yourself.

Power Packed Pixels In Every Package

There has always been a trade-off in this industry between ease of use and power. Our functions do not require a lot of setups, are well-documented, and most of all, thoroughly debugged. Essential Graphics' ease of use stems from our thoughtfulness and not from a lack of power. We explain what we are doing every step of the way. Our support staff consists of the humans who wrote the functions, so we are thoroughly prepared to assist you after the purchase.

Essential Graphics is a trademark of Essential Software



Caveat Emptor

Make no mistake, this is not a package for the "draw a box around the total field" crowd. This library was designed to help the professional C programmer make money and look good.

We've included a complete set of "rubber-banding" functions. One of the most welcome features is the ability to save/restore images in PC Paintbrush format or bit image. World coordinates and view ports aid in programming portability.

We include the ability to manipulate and rotate character fonts and symbols. You can place characters and symbols anywhere on the screen, and use up to eight fonts at one time.

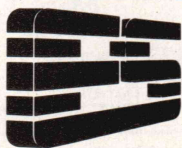
Yours, Mine, Ours

We don't consider ourselves equity partners in your business and therefore we do not charge any royalties or run time fees. We think your efforts belong to you. If for any reason you are unsatisfied with our product you may return it within 30 days for a full refund. Full source is available. Please call today and launch yourself into the world of power graphics.

Price \$299 - Source \$299

Adaptors include - CGA, EGA, VGA, MCGA, ATT, ATT DEB, Hercules, Vega Deluxe, Paradise Autoswitch. **Printer Support** - IBM, Epson, Oki, TI, Alps, Panasonic, and others. **Supports** mice, light pens, plotters, color printers. **Compilers**-Microsoft, Lattice and Turbo-C

Other Essential Products Include: ScreenStar - Essential Communications and Utilities -- /*resident_C*/ - Please call for further information 201-762-6965.



Essential Software, Inc.

South Orange Plaza
76 S. Orange Ave., Suite 3
South Orange, N.J., 07079

**To Order Call:
201-762-6965**

ITALIC FONT IN C
(continued from page 39)

bly language. Ironically, the alleged portability of C code to different compilers (because it is a high-level language) becomes the opposite with TSRs. Because the generated code differs across compilers, code that runs flawlessly compiled with one compiler can produce subtle and hard-to-track bugs on another. The truly "safe" code becomes the one that is "immune" from portability virtues—assembly language.

These objections aside, here is how you implement a simple resident program in Turbo C. Bear in mind that this program does not access the disk or the keyboard, so many TSR techniques are not discussed. For a fuller treatment I recommend *Turbo C: The Art of Advanced Program Design, Optimization and Debugging* by Stephen Randy Davis. I have no doubt that ITALIC could be implemented more efficiently in C than I have done here, but the relevant question is how these improvements compare with the real alternative—assembly language.

First, consider the way that you would implement this TSR in assembly language. Near the top of the source code would be the resident section, preceded by a jump to the installation section, which you would jettison when the resident part was installed. In C you perform the same installation steps of saving the old interrupt 10h vector and installing your own.

A problem arises when you wish to terminate and stay resident. Turbo C contains the *keep()* function, which implements interrupt 21h, function 31h, and requires the number of paragraphs of memory to reserve as one of its parameters. Whereas this is simple to compute in assembly language, it is not so in high-level languages generally or in Turbo C in particular (the Turbo C manual does not even mention this problem).

The method I used in ITALIC was discovered by Dean McCrory and generously posted by him on CompuServe. Turbo C uses two internal variables: `__psp` (to contain the PSP address) and `__brklvl` (to contain

CIRCLE NO. 120 ON READER SERVICE CARD

the address of the end of the initialized and uninitialized data). As memory is dynamically obtained and released, `__brklvl` is adjusted accordingly. If you visualize Turbo C memory allocation as in the diagram for the small model in the *User's Guide* (that is, the data segment follows the code), then adding `__brklvl` to `DS` and subtracting the PSP address gives the size of the code and data. That is what I do in the `keep()` statement at the end of main.

Although this is ingenious, you should be aware of some potential problems and limitations. First, this will not work in the large data models (compact, large, and huge). Second, I do not know what happens, but you must presumably not `farmalloc()` any memory you wish the resident program to use. Third, this is undocumented and should be considered not fully tested.

The other part of the TSR code is the deinstallation routine, including the deallocation of the program's memory. This practice appears to be little known in both C and assembly language (it is certainly not officially documented). It probably enhances the value of a TSR to the user if it is removable, however.

The first thing you do is restore interrupt 10h to its original value. Next, you must deallocate memory. Deallocating the space occupied by a resident program is as easy in C as in assembly language, and the technique is the same. You must remove the program itself and its copy of the environment. In order to do so, when you first load the TSR, you must store the PSP and the contents of the environment pointer, which is located at offset 2ch in the program's PSP.

When you try to install ITALIC, `already_installed()` finds a match and stores the data segment address of the installed copy in the global variable `old_ds`. When you deinstall the program, you peek at `old_ds`, offset by the address of `old_psp` to get the PSP, and then you repeat this using the offset of `old_env` to get the environment address. Next, you deallocate the program memory using interrupt 21h, function 49h. `ES` must hold the address of the installed program's PSP.

Finally, you repeat this function call with `ES` pointing to the installed program's copy of the environment.

It is instructive to run CHKDSK before and after installation to confirm that this procedure works. I have found it to be reliable, but TSRs are a world without rules.

Summary

The techniques for loading custom fonts described here give a program an edge of distinctiveness in an ever more crowded software marketplace. Not only is this now worthwhile for developers because of the greater abundance of EGAs but the code also works on the VGA. This means a fairly long time span for the investment in program development to pay off. Graphics environments offer users custom fonts, but program development is longer. The custom font facilities of the EGA offer a faster development path that does not require the wholesale recoding of applications.

Availability

All the source code for articles in this issue is available on a single

disk. To order, send \$14.95 to Dr. Dobb's Journal, 501 Galveston Dr., Redwood City, CA 94063, or call (415) 366-3600, ext. 216. Please specify the issue number and format (MS-DOS, Macintosh, Kaypro).

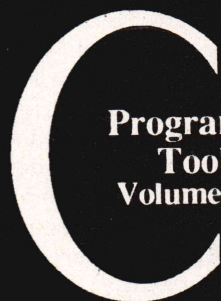
C Windows Toolkit, is a programmers' windowing and video-handling toolkit for Borland International's Turbo C, Mix Power C, and Microsoft C, Versions 4.0, 5.0, and Quick C. C Windows Toolkit offers strong EGA support, including FONTEEDIT, an EGA font editor. It is available from Magna Carta Software.

DDJ

(Listings begin on page 50.)

Vote for your favorite feature/article.
Circle Reader Service No. 2.

What have YOU been missing?



Programmer's Toolbox Volumes I & II

Create better, faster, higher quality and easier to read programs in a fraction of the time. Let your system do the work for you. With 23 powerful, state of the art tools for the IBM PC and compatibles, both beginners and experts will find programming a breeze.

Easy to use. Unlimited program sizes. Online documentation...

- CFlow™:** Determine program hierarchy, external R/T library functions, etc.;
- CLint™:** More rigorously check program syntax;
- CPrint™:** Beautify source programs to user selected formats;
- CXref™:** Cross reference and check symbol usage;
- CritPath™:** Determine a program's critical path;
- PMon™:** Monitor program/OS execution; and more...

At \$79.95 per volume or \$130 for both with a 30 day money back guarantee, the Toolbox is simply the best value today. The Toolbox works with your existing C compiler(s) and enhances your development environment.

Call and Order Today
Visa, MasterCard Accepted



MMC AD Systems
Box 360845 Milpitas, California 95035
(408) 263-0781

"The C Tool Specialists"

CIRCLE NO. 121 ON READER SERVICE CARD

Threaded Binary Trees

Ease of traversal is the major advantage of a threaded binary tree.

by James Mathews

I recently discovered a data structure called a threaded binary tree that combines the quick random lookup qualities of a binary tree with the easy traversal of a linked list. This data structure provided a clean and reasonably elegant solution to an application program I was developing. In this article I describe how to create and traverse threaded binary trees and provide a set of functions written in Microsoft C that illustrate how I implemented this data structure in my application.

I'm the type of programmer who likes to tinker with programs just to see how they work. That particular personality trait recently lead me to work on an interactive spelling checker for the letters and documents I write (it certainly would have been more cost-effective simply to buy a spelling checker).

The spelling checker required a data structure that would allow all the unique words in a document to be identified and then retrieved in sorted order for comparison with the dictionary. I knew that a binary tree would be a fairly efficient way of identifying and sorting the unique

words in a single step, and methods of building and traversing binary trees are well known and documented.

However, I also wanted the ability to scan back and forth interactively in the list of words not found in the dictionary and select whether a particular word was to be added to the dictionary, marked as misspelt, corrected interactively, or simply ignored. And here is where I ran into trouble with a binary tree—all my solutions to moving back and forth interactively to selected nodes in the tree involved more code than seemed necessary or reasonable. At one point I even considered converting the binary tree to a doubly linked list once the (possibly) misspelled words were identified, just to facilitate scanning back and forth over the words.

And then, while browsing through *Fundamentals of Data Structures* by Horowitz and Sahni,¹ I came across a description of threaded binary trees. Great, a threaded binary tree seemed to be just what I needed.

Each node of a binary tree typically contains pointers to left and right child nodes. If a particular node does not have a left or right child, the corresponding pointer has a null value. In fact, it turns out that

slightly more than half of the pointers in any binary tree will be null. For a binary tree with N nodes, there are $2N$ pointers (two per node) and $N+1$ of those pointers will be null (refer to Horowitz and Sahni or another text for a discussion about why $N+1$ pointers are null).

Threaded binary trees differ from other binary trees in that the null pointers to nonexistent left and right child nodes are used as "threads" to point to prior and successor nodes in the tree. Using the (otherwise) null pointers as threads to prior and successor nodes allows a threaded binary tree to be traversed in order almost as easily as a linked list while retaining the quick lookup of a binary tree. Figure 1, page 43, illustrates (a) a standard binary tree and (b) the same tree with the null pointers replaced by threads.

The Code

Listing One, page 58, shows a C language header file (tbtrees.h) that defines a `TREE_NODE` structure containing some of the fields used by my spelling checker application. A node is created in the threaded binary tree for every unique word found in the document being checked. Each `TREE_NODE` contains a pointer to the node's word

James Mathews, Blue Sky Software,
P.O. Box 232, Absecon, NJ 08201.

(stored as a null-terminated string), an integer used to hold flag values, a usage count indicating how many times the word appears in the document, and pointers to left and right child nodes.

When traversing a threaded tree structure, it is necessary to know if a particular pointer field contains the address of a child node or if the pointer field contains a thread to a prior or successor node. The `tbtree.h` header file also defines two flags, `RBIT` and `LBIT`, which indicate if the right and left child pointers contain valid child node addresses or if they contain threads to other nodes. If the `RBIT` flag is set in a node, then that node has a right child node; if `RBIT` is not set, the right child pointer is a thread. Similarly, the `LBIT` flag identifies the usage of the left child pointer.

Notice that the `RBIT` and `LBIT` flags are the only extra storage required in the tree nodes. The threads occupy the right and left child pointers that would otherwise have been left null. Knuth² suggests that even these flags could be eliminated if a child node always resides at a higher memory address than its parent—a child node would have a higher address than the current node, whereas a thread would always contain a lower address. In my application, I chose to play it safe and added the two 1-bit flags. Because I already required a group of other flags for each node, these additional flags did not increase the size of a node.

`tbtree.c` in Listing Two, page 58, shows the routines I created to build and traverse a threaded binary tree. The function `add2tree()` adds a new word to the tree each time it's called provided the word doesn't already exist in the tree. `add2tree()` compares the prospective new word to existing nodes in the tree until it either locates a node already containing that word or it locates a node to which the new word should be added as a right or left child. If the word already appears in the tree, `add2word()` simply increments the word's usage count and exits. If the word needs to be added to the tree, `add2tree()` invokes the `linsert()` or `rinsert()` function to add a node as

a left or right child, respectively.

Functions `linsert()` and `rinsert()` implement the logic required to add a node to a threaded binary tree. Notice that these routines are actually quite simple. To add a new left child node, the new node is given the left child pointer from the parent node, the new node's right child pointer becomes a thread back to the parent, and the new node is linked to the parent as a left child. Function `rinsert()` is an equally simple implementation to add a right child to a node.

I should point out that the `linsert()` and `rinsert()` routines call function `talloc()` to allocate the memory for a new node. The first implemen-

tation of these routines called the standard `malloc()` library routine to allocate memory, but in analyzing the performance of the program on large documents, I found that `malloc()` was by far the largest consumer of CPU time. I therefore created the `talloc()` function to allocate tree nodes more efficiently. For relatively small tree structures, `malloc()` could certainly be used without noticeable degradation.

The functions shown in Listing Two require the presence of a specially initialized root node (although its structure is the same as that of any other node). The `add2tree()`, `linsert()`, and `rinsert()` functions build the threaded binary tree as a left

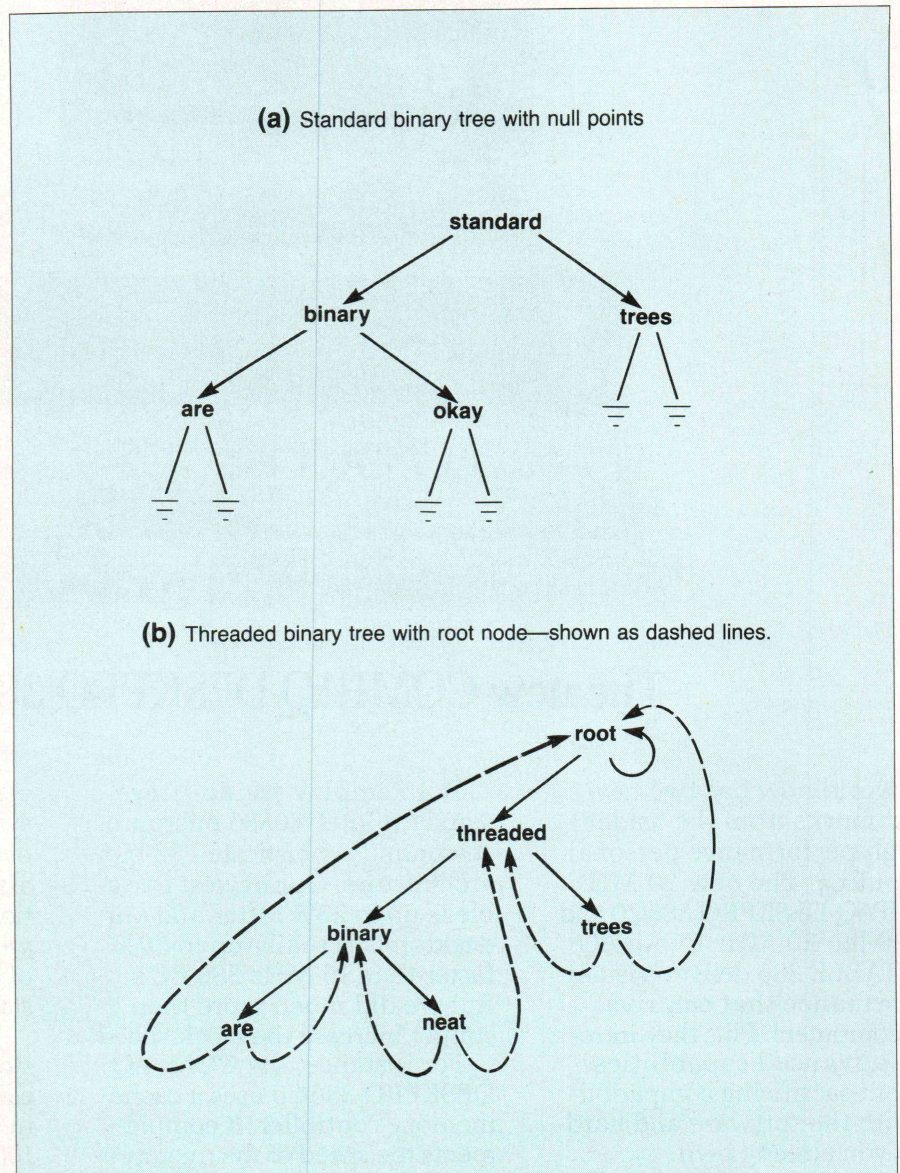


Figure 1: Standard and threaded binary trees

Introducing the two on earth



The new COMPAQ DESKPRO 386/20™

The world now has two new benchmarks from the leader in high-performance personal computing. The new 20-MHz COMPAQ DESKPRO 386/20 and the 20-lb., 20-MHz COMPAQ PORTABLE 386 deliver system performance that can rival minicomputers'. Plus they introduce advanced capabilities without sacrificing compatibility with the software and hardware you already own.

Both employ an industry-standard Intel® 80386 microprocessor and sophisticated 32-bit architecture. Our newest portable is up to 25% faster and our desktop is actually up to 50% faster than 16-MHz 386 PC's. But we did much more than simply increase the clock speed.

For instance, the COMPAQ DESKPRO 386/20 uses a cache memory controller. It complements the speed of the micropro-

cessor, providing an increase in system performance up to 25% over other 20-MHz 386 PC's. It's also the first PC to offer an optional Weitek™ Coprocessor Board, which can give it the performance of a dedicated engineering workstation at a fraction of the cost.

They both provide the most storage and memory within their classes. Up to 300 MB of storage in our latest desktop and up to 100 MB in our new portable.

It simply works better.

most powerful PC's and off.



and the new 20-MHz COMPAQ PORTABLE 386™

Both use disk caching to inject more speed into disk-intensive applications and both will run MS® OS/2.

As for memory, get up to 16 MB of high-speed 32-bit RAM with the COMPAQ DESKPRO 386/20 and up to 10 MB with the COMPAQ PORTABLE 386. Both computers feature the COMPAQ® Expanded Memory Manager, which supports the Lotus®/Intel®/Microsoft® Expanded Memory Specification

to break the 640-Kbyte barrier imposed by DOS.

With these new computers plus the original COMPAQ DESKPRO 386™, we now offer the broadest line of high-performance 386 solutions. They all let you run software being written to take advantage of 386 technology, including Microsoft® Windows/386 Presentation Manager. It provides multitasking capabilities with

today's DOS applications to make you considerably more productive. But that's just the beginning. For more information, call 1-800-231-0900, Operator 43. In Canada, call 416-733-7876, Operator 43.

Intel, Lotus, Microsoft, and Weitek are trademarks of their respective companies.
©1987 Compaq Computer Corporation.
All rights reserved.

COMPAQ®

BINARY TREES

(continued from page 43)

subtree of the root node. An empty threaded binary tree consists of just the root node.

The root node contains a data value that is higher than that of any other node in the tree (I chose the ASCII character as the root node's word value because . is numerically greater than any uppercase or lowercase alphabetic character). This avoids special code in the *add2tree()* function for dealing with the root node.

The right and left child pointers

and the *RBIT* and *LBIT* fields in the root node are initialized such that the root node becomes the in-order predecessor of the lowest-valued tree node and the in-order successor of the highest-valued node. Traversing the tree from start to finish begins and ends at the root node. The definition of the root node occurs in Listing Two just prior to the *add2tree()* function.

Function *inorder_succ()* (also in Listing Two) returns the in-order successor of any node in the threaded binary tree. To find the successor node, *inorder_succ()* looks at the right child of the starting node. If

the right child pointer is a thread, then it points to the successor node and *inorder_succ()* simply returns that node address. If the starting node has an actual right child node, however, the successor is the leftmost child of the starting node's right child. The leftmost child is located by a simple *while* statement that moves down the list of left child pointers.

Function *inorder_pred()* correspondingly locates the in-order predecessor of any node in the tree. *inorder_pred()* returns the starting node's left child pointer (if it's a thread) or the rightmost child of the starting node's left child.

To perform an in-order traversal of the entire threaded binary tree, you need only make successive calls to *inorder_succ()* (for nodes in ascending order by data value) or *inorder_pred()* (descending order by data value). The following few lines of code, for example, would print out all the words in the tree in ascending order:

```
TREE_NODE *tp;
```

```
tp = &root; /* start at root */
while ((tp = inorder_succ(tp)) !=
       &root) /* end at root */
    puts(tp->word);
```

In Summary

Ease of traversal is the major advantage of a threaded binary tree. There are other well-known algorithms for traversing binary trees, but some of those require the use of recursion or an auxiliary stack to maintain the current location within the tree. Recursive algorithms and those that use an auxiliary stack require additional memory above and beyond the requirements of the tree itself. For applications in which the size and structure of the tree is not known beforehand (such as a spelling checker), trying to ensure that sufficient memory exists for the program or auxiliary stack can be difficult. Because threaded binary trees use the otherwise null pointers as threads, this type of tree can be traversed without additional run-time storage.

There are other algorithms that can traverse a binary tree without recursion or the use of an auxiliary stack, but they typically require the

**Now for
VAX C & Sun C**

Add C++ to your favorite C Compiler

- Object-oriented C
- Strong type-checking
- Works with your present C Compiler

DESIGNER C++

BENEFITS:

- Works with the C Compiler you now use
- You can incrementally add C++ features to C (switch-selectable)
- Makes C more suitable for — very large programs — more sophisticated applications
- More reusable code
- Resilient and bug-free code

The only commercially-available C++ customized to operate on PC's, micros, minis, and mainframes with popular C compilers, including:

VAX C
ULTRIX C
APOLLO
XENIX
HP-9000

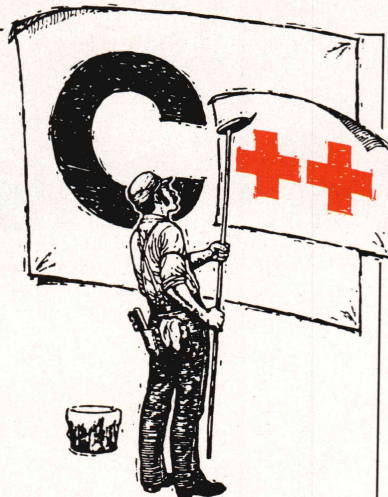
SUN C
MICROSOFT
LATTICE
GREEN HILLS
UNISOFT

*Lattice and Microsoft versions of Designer C++ are known as Advantage C++.

FEATURES:

- Fully compatible with AT&T C++ standard
- Optional strong type checking
- Data abstraction
- Overloading of function names and operators
- Dynamic typing (virtual functions)
- User-defined implicit type conversion
- Works with Sun's *dbxtool*

We Specialize in: Cross/Native Compilers: C, Pascal, FORTRAN, Ada, LISP — Assemblers/Linkers — Symbolic Debuggers — Simulators — Interpreters — Profilers — QA Tools — Design Tools — Comm. Tools — OS Kernels — Editors — VAX & PC Attached Processors and more
We Support: 680xx, 80x86, 320xx, 68xx, 80xx; Clipper, and dozens more



A DIVISION OF XEL

Oasys

230 Second Avenue, Waltham, MA 02154 (617)890-7889

Designer C++ is a joint trademark of XEL, Inc. and Glocksenspiel, Ltd. of Dublin, Ada is a trademark of the U.S. Government (AJPO). Advantage C++ is a trademark of Lifeboat Associates, Inc. Other trademarks are acknowledged to DEC, Lattice, Microsoft & Sun Microsystems, Inc.

CIRCLE NO. 122 ON READER SERVICE CARD

Fast database development system with SQL-based db_QUERY and Lotus 123 interface. . . .TM

db_Vista III

C PROGRAMMERS-

We asked what you wanted in a database development system and we built it!

db_VISTA IIITM is the database development system for programmers who want powerful, high performance DBMS capabilities ... and in any environment. Based on the network database model and the B-tree indexing method, db_VISTA III gives you the most powerful and efficient system for data organization and access. From simple file management to complex database structures with millions of records, db_VISTA III runs on most computers and operating systems like MS-DOS, UNIX, VAX/VMS and OS/2. It's written in C and the complete source code is available, so your application performance and portability are guaranteed! With db_VISTA III you can build applications for single-user microcomputers to multi-user LANs, up to minis and even mainframes.

FAST • PORTABLE • ROYALTY-FREE

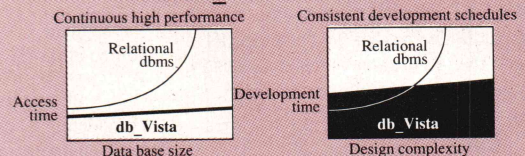
PROFESSIONAL SERVICES: In addition to 60 days of FREE technical support, we offer complete services to get your development project going and keep it on track:

Training Classes • Extended Support • Applications Development & C Programming Services • Consulting • Database Design & Optimization • Product Modification

We're committed to making your database project a success!

HOW TO ORDER: Call us; we'll help determine your needs and get you started. Add components as you need them. Ask about the new Lotus interface. . . Call today!

The db_Vista Difference



The db_VISTA IIITM Database Development System

1 db_VISTATM: The High Performance DBMS

The major features include:

- Multi-user support for LANs and multi-user computers.
- Multiple database access.
- File and record locking.
- Automatic database recovery.
- Transaction processing and logging.
- Timestamping.
- Database consistency check utility.
- Fast access methods based on the network database model and B-tree indexing.
- An easy-to-use interactive database access utility.
- File transfer utilities for importing/exporting ASCII text and dBASE II/III files.
- A Database Definition Language patterned after C.
- Virtual memory disk caching for fast database access.
- A runtime library of over 100 functions.

2 db_QUERYTM: The SQL-based Query.

- Provides relational view of db_VISTA applications.
- Structured Query Language
- C linkable.
- Predefine query procedures or run ad-hoc queries "on the fly".

3 db_REVISETM: The Database Restructure Program.

- Redesign your database easily.
- Converts all existing data to revised design.

4 WKS Library for Lotus 123.

- C-linkable interface to Lotus files.
- **Operating systems:** MS-DOS, UNIX V, XENIX, VMS, OS/2.
- **C Compilers:** Lattice, Microsoft, IBM, Aztec, Computer Innovations, Turbo C, XENIX, and UNIX.
- **LAN systems:** LifeNet, NetWare, PC Network, 3Com, SCO XENIX-NET, other NET- BIOS compatible MS-DOS networks.

All components feature royalty-free run-time distribution, source code availability and our commitment to customer service. That's why corporations like ARCO, AT&T, Hewlett-Packard, IBM, Northwestern Mutual Life, UNISYS and others use our products.

db_VISTA IIITM Database Development System

db_VISTA IIITM
db_QUERYTM
db_REVISETM
db_VISTATM File Manager
WKS Library for Lotus 123

\$595 - 3960
\$595 - 3960
\$595 - 3960
Starts at \$195
Starts at \$195

When high quality data base applications with outstanding performance are important to your company's success:



CALL 1-800-db-RAIMA



(that's 1-800-327-2462)

In the UK call Systemstar Ltd. 0992-400919

RAIMATM
CORPORATION

3055 112th Avenue N.E., Bellevue, WA 98004 (206) 828-4636
Telex: 6503018237MCIUW FAX: (206) 828-3131

Programming Secrets Revealed

What They Don't Teach You About TSR's And ISR's At MIT

When our customers started asking for a resident communications program I thought it would be a piece of cake. I was surprised by how little I really knew about interrupt calls to DOS and their attendant dangers.

Catch-22

The real Catch-22 is that TSR's are multi-programming in a very real sense, and MS-DOS was never presented to the public as multi-anything.

All things considered, they would still be elementary if all you had to do was issue system bios calls, such as interrupts 10h, 14h, 17h, 16h, etc. But, I'm getting ahead of myself.

The secrets of TSR's and interrupts are closely guarded by the companies that develop them. So coming up with any solid information can be frustrating.

Fear Of Terminating

After spending many sleepless nights in front of a CRT using all the tricks of the trade, including help from the Periscope III Debugger, the project was completed. I had lost my fear of terminating and staying resident.

In the process I became familiar with a couple of powerful, undocumented DOS interrupts that Microsoft had tucked away for their own use.

In addition, there were some gross misunderstandings about other functions listed on the better BBS's. Even some of our own functions from Essential Communications had to be wired together in new ways.

So Close I Could Taste It

When I was finally done, I recognized that we were only a short distance away from devising a method for making any C program memory-resident and well-behaved.

Being a red-blooded American Entrepreneur, I was eager to market the fruits of our labor. In fact, by the time we put the final touches on the /*resident_C*/ library, we felt that maybe we should guard the secrets just as closely as everyone else.



The Good Guys Wear White Hats

After heated debate, we decided that publishing a library of interrupts without explaining how and what we did would not be fair to our public. So not only do we supply you with the proper functions, but we go into detail as to the do's and don'ts of memory-resident programming.

/*resident_C*/ will allow you to make any C program a TSR without knowing any of the secret society handshakes (pun intended). However, if you do want to join the club, we provide all the knowledge you'll need.

Make Any C Program Memory Resident

Regardless of what your program does: communications, disk writes and reads, DOS calls, or graphics, you can make it a safe, well-behaved TSR.

/*resident_C*/ will also allow you to create things like memory-resident libraries to be shared by a number of different programs.

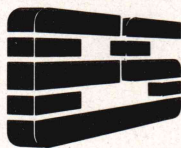
/*resident_C*/ can also be ordered with source code for the really adventurous.

As with any Essential product /*resident_C*/ comes with a 30 day money-back guarantee. Available for Microsoft, Lattice and Turbo C. So call today and establish residency with /*resident_C*/

Price \$99 - Source \$99

Other Essential Products Include

ScreenStar - Essential Communications, Graphics and Utilities - Please call for further information 201-762-6965



**To Order Call:
201-762-6965**

Essential Software, Inc.

South Orange Plaza
76 S. Orange Ave., Suite 3
South Orange, N.J., 07079

/*resident_C*/ is a trademark of Essential Software

BINARY TREES

(continued from page 46)

addition of a parent node pointer field to each node or the temporary reversal of the direction of the pointer fields and the addition of a flag to indicate if a particular node has already been processed. The linked list approach of a threaded binary tree is conceptually simpler and easier to implement.

Threaded binary trees are not a new form of data structure. They were documented in 1960 by Perlis and Thornton³ and are discussed by Knuth² in his *The Art of Computer Programming* series. Threaded binary trees do not seem to have received the widespread recognition or usage that other forms of binary trees have achieved, however. I was prompted to write this article in the hope that I could introduce (or possibly reintroduce) the concepts to others who might find them useful.

Availability

All the source code for articles in this issue is available on a single disk. To order, send \$14.95 to Dr. Dobb's Journal, 501 Galveston Dr., Redwood City, CA 94063, or call (415) 366-3600, ext. 216. Please specify the issue number and format (MS-DOS, Macintosh, Kaypro).

Notes

1. Ellis Horowitz and Sartaj Sahni, *Fundamentals of Data Structures* (Rockville, Md.: Computer Science Press, 1982).
2. D. Knuth, *The Art of Computer Programming: Fundamental Algorithms, Volume 1*, 2d ed (Reading, Mass.: Addison-Wesley, 1973).
3. A. Perlis and C. Thornton, "Symbol Manipulation by Threaded Lists," *Communications of the ACM*, vol. 3, no. 4 (April 1960): 195-204.

DDJ

(Listings begin on page 58.)

Vote for your favorite feature/article.
Circle Reader Service **No. 3.**

The Ada® world just changed. Again.

Meridian continues its tradition of Ada "firsts" by introducing a powerful new set of integrated software development tools.

AdaVantage v2.1 Optimizing Ada Compiler

Available for the IBM PC and compatibles and the Apple Macintosh. An exceptionally fast validated Ada compiler that for the first time brings a true world-class production quality Ada compiler to your desktop personal workstation.

AdaVantage Debugger An interactive source-level debugger for use with programs written using the Meridian AdaVantage compiler. The debugger allows the programmer complete control over the execution of an Ada program in high-level Ada terms — no knowledge of the underlying machine architecture is required. The debugger supports breakpoints, subprogram traces, single-stepping, call backtraces, and full Ada reference syntax.

Ada Developer Interface A powerful interactive screen-oriented interface to the Ada library dependency information. Includes built-in operations to edit, "pretty print", compile, and link any Ada library unit. Configuration file allows user tailoring of all operations and displays.

Run-Time Customization Library For preparation of Ada application programs for execution on 80x86-based embedded systems or other operating systems. The Library is a collection of Ada source files, batch files, and documentation that

define the customizable, system-dependent components of the AdaVantage Run-Time System.

AdaStarter Identical to the validated v2.1 AdaVantage compiler with limitations that permit up to ten library units, each with up to two-hundred executable statements (unlimited declarations and comments). This is approximately equivalent to a 4000 line program. The price of AdaStarter is applicable towards purchase of the AdaVantage production compiler. Get the full power of Ada for only \$99!

AdaDesigner A collection of tools supporting the design, programming, and documentation phases of the software life cycle. Tools include a structured editor/synthesizer coupled to a text editor/incremental syntax analyzer for Ada, an incremental editor for design languages, a program derivation processor that guarantees your design is always in sync with your implementation, and a structured documentation generator.

Configuration The compilers all run in a standard PC configuration with 640K of memory (1MB on the Macintosh) and a hard disk.

For more information call 1-800-221-2522.
In California, call 714-380-9800.

“ **Meridian Software Systems AdaVantage version 2.0 compiler is an Ada software milestone.** ”
— PC Week



23141 VERDUGO DRIVE • SUITE 105 • LAGUNA HILLS, CALIFORNIA 92653
800/221-2522 (Outside California) • 714/380-9800 (Inside California)
Telex: 650-268-0547 MCI • Fax: 714/380-1683

□ The information contained herein is subject to change without notice. □ Ada is a registered trademark of the U.S. Government (AJPO). AdaVantage, AdaTraining, AdaDesigner and AdaStarter are trademarks of Meridian Software Systems, Inc. References to other computer systems use trademarks owned by the respective manufacturers. □ Copyright ©1988 Meridian Software Systems, Inc. All rights reserved.

CIRCLE NO. 124 ON READER SERVICE CARD

ITALIC FONT IN C

Listing One (Text begins on page 36.)

```

/* ITALIC.ASC -- This is an ASCII representation of the italic font */
/* characters used in ITALIC.C. This file is #included. */
/* In the table below, each row corresponds to a character. The */
/* 14 elements of each row correspond to the 14 scan lines of the */
/* character. */
char italic_arr[128-32][14] = {
/* Font character 32 (ASCII value ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 33 (ASCII value ! ) is */ {0x00, 0x00, 0x06, 0x0f, 0x1e, 0x1e, 0x18, 0x18, 0x00, 0x30, 0x60, 0x00, 0x00, 0x00},
/* Font character 34 (ASCII value " ) is */ {0xc0, 0x0c, 0x99, 0x19, 0x12, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 35 (ASCII value # ) is */ {0x00, 0x00, 0x1b, 0x1b, 0x7f, 0x36, 0x6c, 0x6c, 0xf6, 0xd9, 0xb0, 0x01, 0x00, 0x00},
/* Font character 36 (ASCII value $ ) is */ {0x03, 0x03, 0x9f, 0xf1, 0x61, 0xe0, 0x7c, 0x06, 0x0c, 0x8d, 0xf0, 0x61, 0xc0, 0x00},
/* Font character 37 (ASCII value % ) is */ {0x00, 0x00, 0x00, 0x00, 0x61, 0xe3, 0x0c, 0x18, 0x60, 0xcc, 0x18, 0x03, 0x00, 0x00},
/* Font character 38 (ASCII value & ) is */ {0x00, 0x00, 0x0e, 0x1b, 0x36, 0x1c, 0x76, 0xd6, 0x98, 0x99, 0xd8, 0x01, 0x00, 0x00},
/* Font character 39 (ASCII value ' ) is */ {0x00, 0x06, 0x0c, 0x0c, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 40 (ASCII value ( ) is */ {0x00, 0x00, 0x03, 0x06, 0x18, 0x18, 0x30, 0x30, 0x60, 0x30, 0x30, 0x00, 0x00, 0x00},
/* Font character 41 (ASCII value ) ) is */ {0x00, 0x00, 0x0c, 0x06, 0x06, 0x0c, 0x18, 0x30, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 42 (ASCII value * ) is */ {0x00, 0x00, 0x00, 0x00, 0x33, 0x1e, 0xff, 0x3c, 0xcc, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 43 (ASCII value + ) is */ {0x00, 0x00, 0x00, 0x00, 0x0c, 0x0c, 0x7e, 0x18, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 44 (ASCII value , ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 45 (ASCII value - ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 46 (ASCII value . ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 47 (ASCII value / ) is */ {0x00, 0x00, 0x80, 0x01, 0x06, 0xc0, 0x30, 0x60, 0x80, 0x01, 0x00, 0x00, 0x00, 0x00},
/* Font character 48 (ASCII value 0 ) is */ {0x00, 0x00, 0x1f, 0x71, 0x63, 0xc6, 0xf6, 0xe6, 0x8c, 0x8c, 0xf8, 0x00, 0x00, 0x00},
/* Font character 49 (ASCII value 1 ) is */ {0x00, 0x00, 0x06, 0x0e, 0x3c, 0x0c, 0x0c, 0x18, 0x30, 0x30, 0xf8, 0x00, 0x00, 0x00},
/* Font character 50 (ASCII value 2 ) is */ {0x00, 0x00, 0x1f, 0x31, 0x03, 0x06, 0x18, 0x30, 0xc0, 0x8c, 0xf8, 0x00, 0x00, 0x00},
/* Font character 51 (ASCII value 3 ) is */ {0x00, 0x00, 0x00, 0x1f, 0x11, 0x03, 0x03, 0x3c, 0x06, 0xc, 0x8c, 0xf0, 0x00, 0x00, 0x00},
/* Font character 52 (ASCII value 4 ) is */ {0x00, 0x00, 0x03, 0x07, 0x1e, 0x36, 0xcc, 0xf6, 0x18, 0x78, 0x00, 0x00, 0x00, 0x00},
/* Font character 53 (ASCII value 5 ) is */ {0x00, 0x00, 0x3f, 0x30, 0x60, 0x60, 0x7c, 0x06, 0xc, 0x8c, 0xf0, 0x00, 0x00, 0x00},
/* Font character 54 (ASCII value 6 ) is */ {0x00, 0x00, 0x0e, 0x18, 0x60, 0x60, 0xf6, 0xc6, 0x8c, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 55 (ASCII value 7 ) is */ {0x00, 0x00, 0x3f, 0x71, 0x03, 0x06, 0x18, 0x30, 0x60, 0x60, 0xc0, 0x00, 0x00, 0x00},
/* Font character 56 (ASCII value 8 ) is */ {0x00, 0x00, 0x1f, 0x71, 0x63, 0x63, 0x7c, 0xc6, 0x8c, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 57 (ASCII value 9 ) is */ {0x00, 0x00, 0x1f, 0x71, 0x63, 0x63, 0x7e, 0x06, 0xc, 0x18, 0xe0, 0x00, 0x00, 0x00, 0x00},
/* Font character 58 (ASCII value : ) is */ {0x00, 0x00, 0x00, 0x06, 0xc, 0x00, 0x00, 0x00, 0x30, 0x30, 0x00, 0x00, 0x00, 0x00},
/* Font character 59 (ASCII value ; ) is */ {0x00, 0x00, 0x00, 0x06, 0xc, 0x00, 0x00, 0x00, 0x30, 0x30, 0x00, 0x00, 0x00, 0x00},
/* Font character 60 (ASCII value < ) is */ {0x00, 0x00, 0x01, 0x03, 0x0c, 0x18, 0x60, 0x30, 0x30, 0x18, 0x18, 0x00, 0x00, 0x00, 0x00},
/* Font character 61 (ASCII value = ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0x00, 0x00, 0xf6, 0x00, 0x00, 0x00, 0x00},
/* Font character 62 (ASCII value > ) is */ {0x00, 0x00, 0x18, 0x0c, 0x0c, 0x06, 0x06, 0x0c, 0x30, 0x60, 0x80, 0x01, 0x00, 0x00, 0x00},
/* Font character 63 (ASCII value ? ) is */ {0x00, 0x00, 0x9f, 0xf1, 0x63, 0x06, 0x18, 0x18, 0x00, 0x30, 0x60, 0x00, 0x00, 0x00, 0x00},
/* Font character 64 (ASCII value @ ) is */ {0x00, 0x00, 0x9f, 0xf1, 0x63, 0xef, 0xde, 0x8c, 0x8c, 0x8c, 0x8c, 0x00, 0x00, 0x00, 0x00},
/* Font character 65 (ASCII value A ) is */ {0x00, 0x00, 0x04, 0x0e, 0x36, 0x62, 0xc6, 0xf6, 0x8c, 0x8c, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 66 (ASCII value B ) is */ {0x00, 0x00, 0x3f, 0x1b, 0x33, 0x33, 0x7c, 0x66, 0xcc, 0xc6, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 67 (ASCII value C ) is */ {0x00, 0x00, 0x0f, 0x19, 0x61, 0xe0, 0xc0, 0xc0, 0x84, 0xcc, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 68 (ASCII value D ) is */ {0x00, 0x00, 0x3e, 0x1b, 0x33, 0x33, 0x66, 0x66, 0xcc, 0xd8, 0xe0, 0x00, 0x00, 0x00, 0x00},
/* Font character 69 (ASCII value E ) is */ {0x00, 0x00, 0x3f, 0x19, 0x31, 0x30, 0x78, 0x60, 0xc4, 0xcc, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 70 (ASCII value F ) is */ {0x00, 0x00, 0x0f, 0x19, 0x31, 0x30, 0x7c, 0x60, 0xc0, 0xc0, 0xc0, 0x00, 0x00, 0x00, 0x00},
/* Font character 71 (ASCII value G ) is */ {0x00, 0x00, 0x0f, 0x19, 0x61, 0xe0, 0xc0, 0xc0, 0xc0, 0xc0, 0xc0, 0x00, 0x00, 0x00, 0x00},
/* Font character 72 (ASCII value H ) is */ {0x00, 0x00, 0x31, 0x71, 0x63, 0xe3, 0xf6, 0xc6, 0xcc, 0x8c, 0x8c, 0x00, 0x00, 0x00, 0x00},
/* Font character 73 (ASCII value I ) is */ {0x00, 0x00, 0x0f, 0x06, 0xc, 0xc, 0x18, 0x30, 0x30, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 74 (ASCII value J ) is */ {0x00, 0x00, 0x07, 0x03, 0x06, 0x06, 0x0c, 0xc, 0x98, 0x98, 0xe0, 0x00, 0x00, 0x00, 0x00},
/* Font character 75 (ASCII value K ) is */ {0x00, 0x00, 0x39, 0x19, 0x36, 0x36, 0x78, 0x68, 0xc8, 0xc8, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 76 (ASCII value L ) is */ {0x00, 0x00, 0x3c, 0x18, 0x30, 0x30, 0x60, 0x60, 0xc4, 0xc6, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 77 (ASCII value M ) is */ {0x00, 0x00, 0x31, 0x7b, 0x7f, 0xf6, 0xd6, 0xc6, 0x8c, 0x8c, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 78 (ASCII value N ) is */ {0x00, 0x00, 0x31, 0x79, 0x7b, 0xff, 0xde, 0xc6, 0x8c, 0x8c, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 79 (ASCII value O ) is */ {0x00, 0x00, 0x0e, 0x1b, 0x63, 0xe3, 0xc6, 0xc6, 0x8c, 0xd8, 0xe0, 0x00, 0x00, 0x00, 0x00},
/* Font character 80 (ASCII value P ) is */ {0x00, 0x00, 0x3f, 0x19, 0x33, 0x33, 0x7c, 0x60, 0xc0, 0xc0, 0xc0, 0x80, 0x00, 0x00, 0x00},
/* Font character 81 (ASCII value Q ) is */ {0x00, 0x00, 0x1f, 0x71, 0x63, 0xe3, 0xc6, 0xd6, 0xc6, 0xc0, 0xf8, 0x30, 0x38, 0x00, 0x00},
/* Font character 82 (ASCII value R ) is */ {0x00, 0x00, 0x3f, 0x13, 0x33, 0x33, 0x7c, 0x6c, 0xcc, 0xc6, 0xf8, 0x00, 0x00, 0x00, 0x00},
/* Font character 83 (ASCII value S ) is */ {0x00, 0x00, 0x1f, 0x71, 0x63, 0x30, 0x38, 0xc0, 0xc6, 0xc6, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 84 (ASCII value T ) is */ {0x00, 0x00, 0x3f, 0x3f, 0x2d, 0xc0, 0x18, 0x18, 0x30, 0x60, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 85 (ASCII value U ) is */ {0x00, 0x00, 0x31, 0x71, 0x63, 0xe3, 0xc6, 0xc6, 0xc6, 0xc6, 0xf8, 0x00, 0x00, 0x00, 0x00},
/* Font character 86 (ASCII value V ) is */ {0x00, 0x00, 0x31, 0x71, 0x63, 0xe3, 0xc6, 0xc6, 0xd8, 0xf0, 0xc0, 0x00, 0x00, 0x00, 0x00},
/* Font character 87 (ASCII value W ) is */ {0x00, 0x00, 0x31, 0x71, 0x63, 0xc3, 0xd6, 0xd6, 0xd6, 0xf8, 0xb0, 0x00, 0x00, 0x00, 0x00},
/* Font character 88 (ASCII value X ) is */ {0x00, 0x00, 0x31, 0x71, 0x36, 0x1c, 0x38, 0x78, 0xd8, 0x8c, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 89 (ASCII value Y ) is */ {0x00, 0x00, 0x1b, 0x1b, 0x32, 0x36, 0x3c, 0x18, 0x30, 0x30, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 90 (ASCII value Z ) is */ {0x00, 0x00, 0x3f, 0x71, 0x46, 0xc0, 0x30, 0x60, 0x84, 0x8c, 0xf8, 0x00, 0x00, 0x00, 0x00},
/* Font character 91 (ASCII value [ ) is */ {0x00, 0x00, 0x0f, 0xc, 0x18, 0x30, 0x30, 0x60, 0x60, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 92 (ASCII value \ ) is */ {0x00, 0x00, 0x20, 0xf0, 0x70, 0x38, 0x38, 0x1c, 0xc, 0xc, 0x08, 0x00, 0x00, 0x00, 0x00},
/* Font character 93 (ASCII value ] ) is */ {0x00, 0x00, 0x0f, 0xc3, 0x06, 0x06, 0xc, 0xc, 0x18, 0x18, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 94 (ASCII value ^ ) is */ {0x02, 0x07, 0x9b, 0xf1, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 95 (ASCII value _ ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 96 (ASCII value ` ) is */ {0x06, 0x06, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x00, 0x00},
/* Font character 97 (ASCII value a ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 98 (ASCII value b ) is */ {0x00, 0x00, 0x38, 0x18, 0x30, 0x3c, 0xc6, 0x66, 0xcc, 0xc6, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 99 (ASCII value c ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3e, 0xc6, 0xc0, 0x80, 0x8c, 0xf0, 0x00, 0x00, 0x00},
/* Font character 100 (ASCII value d ) is */ {0x00, 0x00, 0x07, 0x03, 0x06, 0x1e, 0x6c, 0xcc, 0x98, 0xf8, 0xd8, 0x00, 0x00, 0x00, 0x00},
/* Font character 101 (ASCII value e ) is */ {0x00, 0x00, 0x00, 0x00, 0x3e, 0xc6, 0xf6, 0x80, 0x8c, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 102 (ASCII value f ) is */ {0x00, 0x00, 0x0e, 0x1b, 0x32, 0x30, 0xf8, 0x60, 0xc0, 0xc0, 0xc0, 0x00, 0x00, 0x00, 0x00},
/* Font character 103 (ASCII value g ) is */ {0x00, 0x00, 0x00, 0x00, 0x3b, 0xc6, 0xcc, 0x98, 0xf8, 0x30, 0x30, 0xc0, 0x00, 0x00},
/* Font character 104 (ASCII value h ) is */ {0x00, 0x00, 0x38, 0x18, 0x30, 0x36, 0x76, 0x66, 0xcc, 0xc6, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 105 (ASCII value i ) is */ {0x00, 0x00, 0x06, 0x06, 0x00, 0xc, 0x18, 0x18, 0x30, 0x30, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 106 (ASCII value j ) is */ {0x00, 0x00, 0x01, 0x01, 0x00, 0x07, 0x06, 0x06, 0xc, 0xc, 0x98, 0x98, 0xe0, 0x00, 0x00},
/* Font character 107 (ASCII value k ) is */ {0x00, 0x00, 0x38, 0x18, 0x30, 0x33, 0x6c, 0x78, 0xd8, 0xc6, 0x98, 0x00, 0x00, 0x00, 0x00},
/* Font character 108 (ASCII value l ) is */ {0x00, 0x00, 0x0e, 0x06, 0xc, 0xc, 0x18, 0x18, 0x30, 0x30, 0xf0, 0x00, 0x00, 0x00, 0x00},
/* Font character 109 (ASCII value m ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0xf6, 0xf6, 0x66, 0x66, 0xcc, 0xc6, 0x98, 0x00, 0x00, 0x00},
/* Font character 110 (ASCII value n ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0xf6, 0xf6, 0x66, 0x66, 0xcc, 0xc6, 0x98, 0x00, 0x00, 0x00},
/* Font character 111 (ASCII value o ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 112 (ASCII value p ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 113 (ASCII value q ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 114 (ASCII value r ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 115 (ASCII value s ) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3e, 0x46, 0x70, 0x38, 0x8c, 0xf0, 0x00, 0x00, 0x00}

```

(continued on page 52)

Only WATERLOO C's programming environment delivers all 10 of the most wanted user features:

- ✓ High-speed compilation
- ✓ High-performance code
- ✓ Reentrant code options
- ✓ OS linkage options
- ✓ In-line functions
- ✓ ANSI compiler and library
- ✓ Full-screen library
- ✓ Full-screen symbolic debugger
- ✓ Extensive on-line documentation
- ✓ Development tools

WATERLOO C Version 3.0 is the high-performance development system for IBM 370 mainframe architectures under VM/CMS consisting of:

- **Optimizing C Compiler**
- **C Run-time Library**
- **Interactive Source-level Debugger**
- **Development Tools**

WATERLOO C is ideal for new software development and for porting existing software to IBM mainframes. It provides ANSI standard support in an integrated CMS environment for portability and efficiency. And it puts all the resources of a full development system at your fingertips so you can quickly achieve quality results.

WATERLOO C improves your response time and conserves your mainframe resources with an extremely fast compilation rate. A single phase from source to object means less overhead, while both the compiler and library can reside in shared segments for even faster compiles and links.

WATERLOO C generates the highest quality code with an exclusive state-of-the-art optimizer. Options for reentrant applications and OS linkage conventions mean flexibility in development. Moreover, a convenient option allows you to see the original source lines as comments in the assembler output.

The source-level debugger lets you work with your programs the way you wrote them. The full-screen interface feels so natural that it is easy to use. Debugging activity is performed using source variables, functions, and lines. And the execution environment allows stepping, breakpoints, tracing, and display and alteration of your program data.



```

118 char *question_macro, *comm;
119 register retval = FALSE;
120
121 GetlineStart( root );
122 newer = OutDate( root, EXECUTEALL );
123 if ( ( QUESTION == TRUE ) && ( Updatit
124 (
125     FreelistItems( newer );
126     retval = TRUE;
127     return( retval );
128 )
129 question_macro = QuesMacro( newer );
130 if( (question_macro != NULL) || (EXECUTEALL == TRUE) )
131 (
132     if( TOUCH == TRUE )
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

WATERLOO C 3.0 full-screen debugger: A quick route to quality results.

Completing the development environment are these important productivity tools:

- Update your programs with a single command: **MAKE**.
- Simplify multi-file maintenance with **GREP** and **DIFF**.
- Track source variables and functions with cross-reference utilities.
- Spot performance bottlenecks with a program execution profiler.

A full support framework provides complete technical backup. Access documentation easily on-line or in comprehensive reference manuals. Our Customer Support Center provides you with expert technical assistance. Newsletters keep you informed about product developments. And on-site installation and training workshops are available.

Discover the high-performance advantage of WATERLOO C. Write or call for our complete information package:

WATCOM

Find out how quickly and easily you can generate the highest quality code for IBM mainframes by writing or calling for the WATERLOO C complete product information package.

- ☐ Please send your product information package.
- ☐ I would like to arrange for an evaluation copy.

Name: _____
 Title: _____
 Company: _____
 Street: _____
 City: _____
 State: _____ Zip: _____
 Telephone: _____

WATCOM

Dept. DD03A
 415 Phillip Street
 Waterloo, Ontario, Canada N2L 3X2
Tel. (519) 886-3700

ITALIC FONT IN C

Listing One (Listing continued, text begins on page 34.)

```
/* Font character 116 (ASCII value t) is */ {0x00, 0x00, 0x04, 0x0c, 0x18, 0x7e, 0x30, 0x30, 0x60, 0x6c, 0x70, 0x00, 0x00, 0x00},
/* Font character 117 (ASCII value u) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0xe6, 0xcc, 0xcc, 0x98, 0x98, 0xd8, 0x00, 0x00, 0x00},
/* Font character 118 (ASCII value v) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x66, 0x66, 0x66, 0xc8, 0x60, 0x00, 0x00, 0x00},
/* Font character 119 (ASCII value w) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0x46, 0xd6, 0xac, 0xfc, 0xb0, 0x00, 0x00, 0x00},
/* Font character 120 (ASCII value x) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0x6c, 0x38, 0x70, 0xd8, 0x98, 0x00, 0x00, 0x00},
/* Font character 121 (ASCII value y) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0x66, 0xc6, 0x8c, 0xfc, 0x18, 0x30, 0xe0, 0x00},
/* Font character 122 (ASCII value z) is */ {0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0x4c, 0x18, 0x60, 0xcc, 0xf8, 0x00, 0x00, 0x00},
/* Font character 123 (ASCII value {) is */ {0x00, 0x00, 0x03, 0x06, 0x0c, 0x0c, 0x70, 0x18, 0x30, 0x30, 0x38, 0x00, 0x00, 0x00},
/* Font character 124 (ASCII value |) is */ {0x00, 0x00, 0x06, 0x06, 0x0c, 0x0c, 0x00, 0x18, 0x30, 0x30, 0x60, 0x00, 0x00, 0x00},
/* Font character 125 (ASCII value }) is */ {0x00, 0x00, 0xc, 0x06, 0x0c, 0x0c, 0x0e, 0x18, 0x30, 0x30, 0xc0, 0x01, 0x00, 0x00},
/* Font character 126 (ASCII value ~) is */ {0x00, 0x00, 0x1d, 0xf7, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
/* Font character 127 (ASCII value ) is */ {0x00, 0x00, 0x00, 0x00, 0x10, 0x38, 0x6c, 0xc6, 0xc6, 0xfe, 0x00, 0x00, 0x00, 0x00}
); /* End of italic array */
```

End Listing One

Listing Two

```
/* Copyright (C) Magna Carta Software, 1987. All Rights Reserved. */
/* MAKEFONT -- Makes an italic font for the EGA. */
/* Note: Do not run this program from within the IDE. */
/* First version 10/29/87. Last update: 10/31/87. */

#ifdef __SMALL__
#error Should use SMALL compilation model
#endif

#include <stdio.h>
#include <dos.h>
#include <process.h>
#include <mem.h>
#include <stdlib.h>

#define TRUE 1
#define FALSE 0

/* Functions related to video operations */
void load_user_egafont(char *fptr, int block, int bpc, int char_count, int spos);
void get_egafont(char *fptr, int font);
int get_video_info(void);

/* Global variables and #DEFINES related to video operations */
#include "mk_ital.asc" /* ITALIC.ASC contains our italic font */
```

Query+MANAGER= Q-MAN

Q-MAN, a fast true relational data base management system which supports interactive and imbedded queries, library routines, forms, and will be multi-user or distributed. Uses QUEL and SQL languages, B+ trees, and sort-merge joins. Free multi-user upgrade when released first quarter of 1988.

Runs on:	single-user
SYS 5	\$1395
4.2	1395
MS-DOS™	329

To order
Call collect

(608) 271-2171

Breakpoint Computer Systems, Inc.
6701 Seybold Road, Suite 204
Madison, WI 53719

Include machine name and operating system
when ordering.

Visa and Mastercard accepted.
MS-DOS™ is a registered trademark of Microsoft Corporation.

CIRCLE NO. 195 ON READER SERVICE CARD

MICROSOFT, TURBO AND MIX POWER C PROGRAMMERS... C WINDOWS TOOLKIT PUTS YOU IN CHARGE OF VIDEO!

C Windows Toolkit is the only C programmer's windowing package that comes with a complete tutorial on monochrome, Hercules, CGA and EGA video. We don't just provide the functions, we also explain how to use them reliably.

And C Windows Toolkit comes with full, commented source code (would you trust a package that didn't?).

WINDOWING FUNCTIONS

- Create pop-up windows
- Create pull-down menus
- Create spreadsheet menus
- Create context-sensitive help screens
- Store windows for recall later
- Free memory used by windows
- Use 8 different types of exploding windows

SYSTEM SUPPORT

- Detect how many video adaptors are present
- Detect the types of video adaptor installed
- Switch between adaptors
- Detect ANSI.SYS
- Control the size and position of the cursor
- Detect the Enhanced Keyboard
- Disable the video signal
- Delay program execution to microsecond resolution

EGA/VGA SUPPORT

- Use all 64 EGA colors
- Use EGA 43-line mode
- Use 2 fonts simultaneously
- Design custom fonts
- Smooth scroll the screen
- Smooth pan the screen

FAST SCREEN I/O

- Write to the screen lightning fast
- Write formatted output (like printf())
- Get snow-free output on the CGA
- Scroll the screen
- Read characters off the screen

HERCULES SUPPORT

- Detect the presence of a Hercules Card
- Detect Ramfont support
- Load a Ramfont
- Switch between modes

Over 80 functions that enhance your productivity.
Full source code included — No run-time royalties
Includes 200 page manual — 30-Day Money-Back Guarantee

**Magna
Carta
SOFTWARE**

Requires: IBM PC, XT, AT, PS/2 or compatible that will run
Microsoft C and/or Quick C
Supports Microsoft C 4.0/5.0/Quick C, Borland Turbo C 1.0/1.5,
Mix Power C

From: **Magna Carta Software**
P.O. Box 475594
Garland, TX 75047-5594
(214) 226-6909



Only \$99.95

(Texas residents add 8% sales tax)

CIRCLE NO. 196 ON READER SERVICE CARD


```

#define VIDEO 0x10                                /* BIOS video interrupt */
char fontarray[3585];                             /* buffer for font storage */
int our_id1 = 0xfac, our_id2 = 0x1000;             /* our ID words */
char ega_color;

/* Functions related to TSR operations */
int already_installed(void);
int deinstall(void);
void interrupt (*old_int10h)(void);
void interrupt int10h(unsigned bp, unsigned di, unsigned si,
                      unsigned ds, unsigned es, unsigned dx,
                      unsigned cx, unsigned bx, unsigned ax);

/* Global variables related to TSR operations */
unsigned save_bp1, save_bp2, old_ds, old_psp;
unsigned old_env;

/* Turbo C system variables (see text for explanation) */
extern unsigned _brklvl;
extern unsigned _psp;

/* Other functions */
void error(int errnum);

main()
{
    int i, j, k;
    union REGS regs;

    get_video_info();
    if (!ega_color) regs.x.ax = 0x7;             /* set the video mode */
    else regs.x.ax = 0x3;
    int86(VIDEO, &regs, &regs);
    if (already_installed()) {
        deinstall();
        printf("\nThe Italic font is now no longer installed");
        exit(0);
    }

    /* system checks out -- go ahead and put italic chars. in font */
    get_egafont(fontarray, 14);                 /* store the ROM font in fontarray */
    for(i=14*32, j=0; i<14*128; j++) {
        for(k=0; k<14; k++) fontarray[i++] = italic_arr[j][k];
    }
    load_user_egafont(fontarray, 0, 14, 256, 0); /* load our font */

    old_psp = _psp;                             /* save the resident program's PSP */
    old_env = peek(_psp, 0x2c);                 /* save the resident program's ENV */
    old_int10h = getvect(VIDEO);               /* save the old vector */
    setvect(VIDEO, int10h);                    /* install our INT10h handler */

    /* terminate and stay resident. Program length is determined by */
    /* subtracting the psp address (_psp) from _brklvl which is */
    /* dynamically set to the address of the end of DS. This */
    /* appears to be reliable in the TINY and SMALL models, but */
    /* results are unknown for other models. */
    keep(FALSE, _DS + (_brklvl + 15) / 16 - _psp);

}

/* ALREADY_INSTALLED: This routine scans through memory for our ID byte. */
/* Returns: 0 if not found, 1 if found. */
int already_installed()
{
    unsigned int next_seg;

    for(next_seg = _psp-1; next_seg > 0; next_seg--) {
        if (peek(next_seg, (unsigned) &our_id1) == our_id1) {
            if (peek(next_seg, (unsigned) &our_id2) == our_id2) {
                old_ds = next_seg;
                return (1);
            }
        }
    }
    return (0);
}

/* DEINSTALL: Remove our TSR by resetting interrupt 0x10 and video mode. */
int deinstall()
{
    union REGS regs;
    struct SREGS sregs;

    /* initialize old interrupt vector */
    old_int10h = MK_FP(peek(old_ds, (unsigned) &old_int10h+2), peek(old_ds, (unsigned)
                                                                    &old_int10h));
    setvect(VIDEO, old_int10h);                /* reset the interrupt vector */

```

(continued on page 55)

It's Our Objective to Keep You Oriented

JOURNAL OF OBJECT-ORIENTED Programming

Editor: Dr. Richard Wiener
Univ. of Colorado

Assoc. Editor: Prof. David Thomas
Carleton Univ.

The Journal of Object-Oriented Programming (JOOP) covers current research, developments and applications in the object-oriented programming (OOP) field.

It features carefully chosen peer-reviewed research articles and quality tutorial papers dealing with OOP.

Six times a year, you will receive well-researched articles on topics useful in your daily work such as:

- ☐ problem solving
- ☐ reusable OOP components
- ☐ language developments
- ☐ software management
- ☐ applications on artificial intelligence
- ☐ program testing
- ☐ software maintenance
- ☐ unbiased product reviews
- ☐ up-to-the minute product news

JOOP maintains a prestigious international editorial board from industry and academia such as:

Adele Goldberg —

Parc Place Systems (Small Talk)

Brad Cox —

PPI, (Objective C)

Bjarne Stroustrup —

AT&T Bell Labs (C++)

Get original research that's definitive, authoritative and applicable. Get objective coverage that's enlightening, reliable and functional.

JOOP is the single communication vehicle uniting all professionals working in the field.

**Order before May 1, 1988,
and become a
CHARTER SUBSCRIBER
for only \$39 and save \$10
off the regular individual rate.**

(outside the U.S. — add \$24)

1 YEAR (six issues) — NOW \$39

Order Toll Free: 1-800-345-8112

The Journal of
Object-Oriented Programming
310 Madison Ave, Suite 503
New York, NY 10017

CIRCLE NO. 127 ON READER SERVICE CARD

DEBUGGING SWAT TEAM

*Order Eco-C88 Rel. 4.0 New Modeling Compiler
and get C-more at no extra charge!*

Seek and Correct

You already know that fast compilation does not mean fast program development. Backing up for bogus error messages and removing the bugs takes time. Eco-C88's "Seek and Correct" three-way error checking finds even the most elusive bugs, clearing the path for swift program development.

Double Barrel Error Checking

Eco-C88 nails **syntax errors** cold and tells you about the error in plain English. And there's no avalanche of false error messages, either. Other compilers can generate up to four times the number of error messages actually present; they leave it up to you to guess which ones are real. You'll be more productive with Eco-C88 because there is no guess work.

Eco-C88 provides ten levels of **semantic error** checking. You can select from almost no checking to the fussiest you've ever seen. Eco-C88's "picky flag" finds subtle errors that slip by other compilers.

Eco-C88 also features:

- All data types, plus ANSI Enhancements
- Robust library, including many new ANSI functions
- CED editor with online function help, split windows, compile-edit-link capability
- New, expanded manual with sample programs for the library functions

C-more Source Code Debugger

Finally, if a really nasty bug persists, put C-more, our source code debugger, to work. With C-more you can watch your program as it executes, single-step it, set simple or conditional breakpoints, test complex expressions, use variables as indexes into other variables, initialize and trace variables, examine CPU registers, display results with printf()-type options and much more. C-more can help you track down bugs in minutes rather than days.

The price for Eco-C88 is \$99.95. And, for a limited time, we'll give you our C-more debugger at no extra charge.

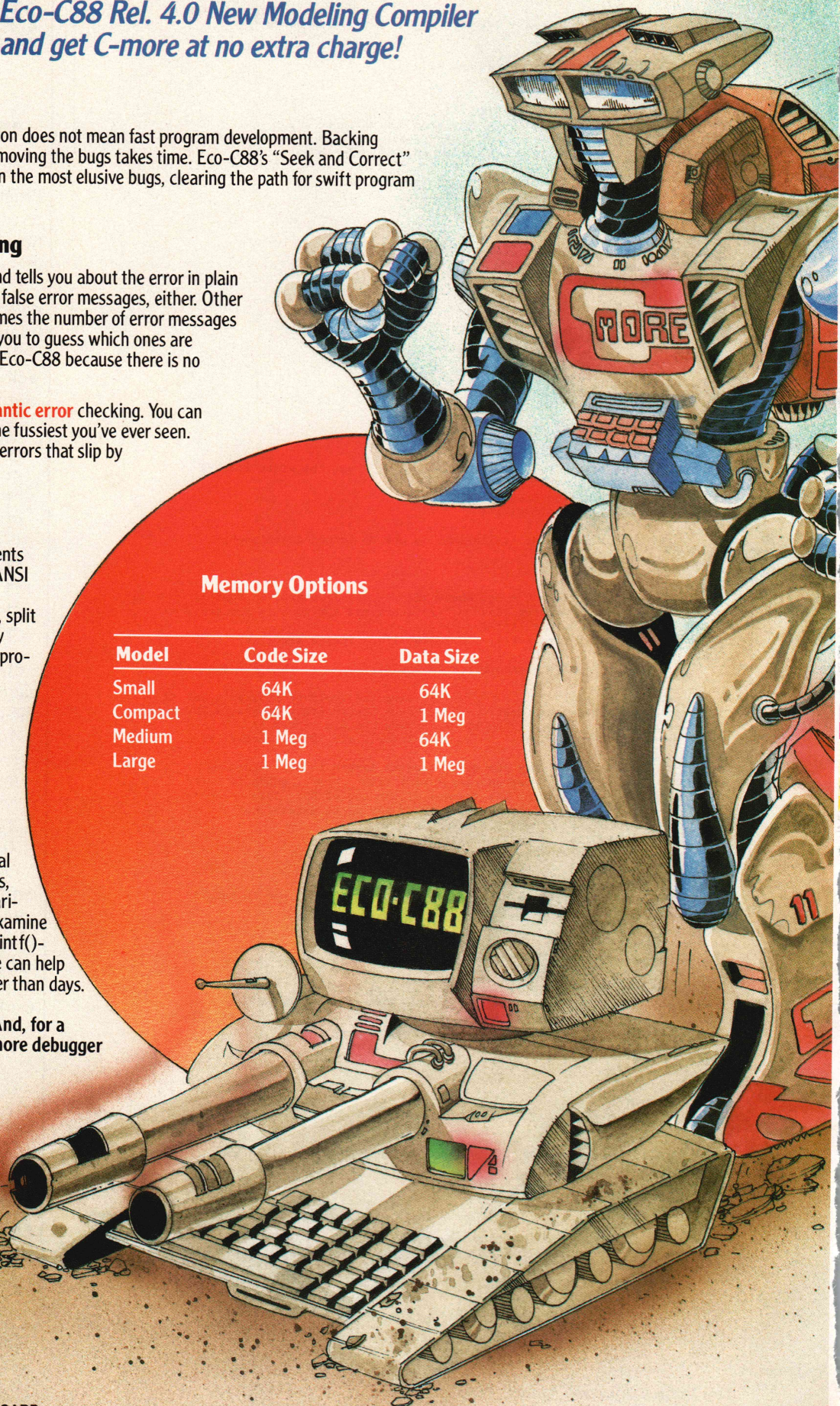
Ecosoft Inc.

6413 N. College Ave.
Indianapolis, IN 46220

(317) 255-6476 (Tech Info)
(800) 952-0472 (Orders)

Memory Options

Model	Code Size	Data Size
Small	64K	64K
Compact	64K	1 Meg
Medium	1 Meg	64K
Large	1 Meg	1 Meg



ITALIC FONT IN C

Listing Two (Listing continued, text begins on page 34.)

```

if (!ega_color) regs.x.ax = 0x7;          /* reset the video mode */
else regs.x.ax = 0x3;
int86(VIDEO,&regs,&regs);

/* Deallocate the memory used by the resident program */
old_psp = peek(old_ds, (unsigned) &old_psp);
old_env = peek(old_ds, (unsigned) &old_env);

regs.x.ax = 0x4900;      /* DOS function to free allocated memory */
sregs.es = old_psp;
intdosx(&regs,&regs,&sregs);
if (regs.x.cflag) error(3);

regs.x.ax = 0x4900;      /* DOS function to free allocated memory */
sregs.es = old_env;
intdosx(&regs,&regs,&sregs);
if (regs.x.cflag) error(4);
return (0);
}

/* INT10: This function is the BIOS video interrupt handler. */
void interrupt int10h(unsigned bp, unsigned di, unsigned si,
    unsigned ds, unsigned es, unsigned dx,
    unsigned cx, unsigned bx, unsigned ax)
{
    /* execute the old video interrupt */
    if (ax >> 8) {          /* it is not a video mode reset */
        _AX = ax;
        _BX = bx;
        _CX = cx;
        _DX = dx;
        save_bp1 = _BP;
        _BP = bp;
        (*old_int10h)();
        _BP = save_bp1;
        ax = _AX;
        bx = _BX;
        cx = _CX;
        dx = _DX;
    }
    else {                  /* AH == 0 for a video mode reset */
        _AX = ax;
        _BX = bx;
        _CX = cx;
        _DX = dx;
        (*old_int10h)();
        ax = _AX;
        bx = _BX;
        cx = _CX;
        dx = _DX;

        /* reload our font destroyed by the mode reset */
        load_user_egaxfont(fontarray,0,14,256,0);
    }
}

/* LOAD_USER_EGAXFONT -- Load a user-defined font and reset page length.*/
/* Params: ptr. to user table, block to load, bytes-per-char, */
/* number of chars to store, starting position in font table. */
/* First version 7/13/87. Last update 10/31/87. */
void load_user_egaxfont(char *fptr,int block,int bpc,int char_count,int spos)
{
    unsigned byte_block;
    byte_block = (bpc << 8) | block;

    /* Can't use intr() due to Turbo C v1.0 compiler bug. */
    /* Note: we must do any assignments to segments prior to doing */
    /* assignments to AX since AX is destroyed. */
    _ES = _DS;
    _AX = 0x1100;          /* call function 0x11 */
    _BX = byte_block;      /* block to load */
    _CX = char_count;      /* number of characters to load */
    _DX = spos;            /* character offset into table */
    save_bp2 = _BP;        /* save BP for stack addressing */
    _BP = FP_OFF(fptr);    /* load address of user font */
    geninterrupt(VIDEO);
    _BP = save_bp2;        /* restore BP -- or die... */
}

/* GET_EGAFONT: This routine grabs an EGA font from ROM and stores it */
/* in the global variable fontarray */
void get_egafont(char *fptr, int font)
{
    struct REGPACK regs;

    regs.r_ax = 0x1130;     /* EGA BIOS call to return font */
    if (font == 8) regs.r_bx = 0x0300;
    else if (font == 14) regs.r_bx = 0x0200;
}

```

(continued on next page)



The C Store™

EVERYTHING FOR THE C PROGRAMMER

PLUS FREE SHIPPING! FREE SOFTWARE!

LATTICE C COMPILER Ver 3.2 The classic DOS development environment.	\$229
MICROSOFT C COMPILER Ver. 5.0 Innovative CodeView™ debugger, "make", more.	\$269
TURBO C COMPILER Ver. 1.0 Fast, full development environment bargain.	\$69
INSTANT C INTERPRETER Ver. 3.0 Instant linking, execution and debugging! Directly link Microsoft, Lattice libraries.	\$379
C-TERP C INTERPRETER Ver. 3.0 Virtual memory support, versions for all compilers.	\$229
PC LINT Ver. 2.13 Shake out C bugs before compiling; neat!	\$99
ESI RESIDENT C Ver. 1.0	\$79
ESI RESIDENT C (w/source) Create TSR programs, handle interrupts!	\$149
PANEL PLUS Ver. 1.0 (w/source) Complete screen I/O development, no royalty.	\$379
WINDOWS FOR C Ver. 4.14	\$149
WINDOWS FOR DATA Ver. 2.06 Flexible, fast, high quality windowing system.	\$229
ESI SCREENSTAR Ver. 1.01	\$79
ESI SCREENSTAR (w/source) Generates C code for windows, data I/O.	\$149
GREENLEAF LIBRARIES:	
Functions Ver. 3.10	\$129
Communications Ver. 2.10	\$129
Data Windows Ver. 2.10	\$159
Data Windows/With Source	\$269
Seasoned, reliable library leader of the pack.	
CTREE Ver. 4.1	\$299
RTREE Ver. 1.1	\$229
CTREE/RTREE Package One of the fastest B-trees, handles networks.	\$499
BTRIEVE Ver. 4.1	\$179
XTRIEVE Ver. 3.02	\$179
XTRIEVE report option Ver. 3.02 Innovative performer, fault tolerant B-tree.	\$99
dbVISTA with Source Ver. 2.21	\$389
dbQUERY with Source Ver. 1.0 Very fast portable B-tree, SQL query option.	\$389
NORTON GUIDE C Ver. 1.0	\$69
NORTON GUIDE C/ASM Twin Pack Ver. 1.0 Online help and reference when you need it.	\$110

THE BEST QUALITY C PRODUCTS AT THE BEST PRICES!



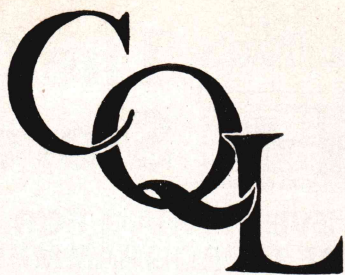
**ORDER TOLL FREE:
(800) 356-0909**

**IN NEW YORK CALL:
(800) 341-1950, EXT. 889**

- **FREE! PC-Write V2.71** complete word processor or **Spectacular Two Player, Real-Time SPACEWAR V1.71** with every order!
- **FREE! No charge for UPS Ground.**
- No surcharge for VISA or Mastercard.
- 24 hour 1200 Baud order line! (914) 241-9324.
- Customer/Technical Support (914) 666-8119.
- No APO, FPO or international orders.
- 30 day money back guarantee on unused items with intact seals.
- Returns subject to 20% restocking fee.
- Call first for RMA Number.

487 East Main Street, Mt. Kisco, NY 10549-0110

CIRCLE NO. 129 ON READER SERVICE CARD



SQL Compatible Query System adaptable to any operating environment.

CQL Query System. A subset of the Structured English Query Language (SEQUEL, or SQL) developed by IBM. Linked files, stored views, and nested queries result in a complete query capability. File system interaction isolated in an interface module. Extensive documentation guides user development of interfaces to other record oriented file handlers.

Portable Application Support System

Portable Windowing System. Hardware independent windowing system with borders, attributes, horizontal and vertical scrolling. User can construct interface file for any hardware. Interfaces provided for PC/XT/AT (screen memory interface and BIOS only interface), MS-DOS generic (using ANSI.SYS), Xenix (both with and without using the curses interface), and C-library (no attributes).

Screen I/O, Report, and Form Generation Systems. Field level interface between application programs, the Query System, and the file system. Complete input/output formatting and control, automatic scrolling on screens and automatic pagination on forms, process intervention points. Seven field types: 8-bit unsigned binary, 16 bit signed binary, 16 bit unsigned binary, 32 bit signed binary, monetary (based on 32 bit binary), string, and date.

Including Source Code

\$395.00

File System interfaces include C-tree and BTRIEVE.

HARDWARE AND FILE SYSTEM
INDEPENDENT

MACHINE INDEPENDENT SOFTWARE CORPORATION

1415 NORTHGATE SQUARE #21D
RESTON, VA 22090

VISA/Master Charge accepted
(703) 435-0413

*C-tree is a trademark of FairCom

IBM, SEQUEL, PC, XT, AT are trademarks of IBM Corp.
MS-DOS and Xenix are trademarks of Microsoft Corp.

CQL and the CQL logo are trademarks of
Kurtzberg Computer Systems.

CIRCLE NO. 130 ON READER SERVICE CARD

ITALIC FONT IN C

Listing Two (Listing continued, text begin on page 34.)

```

    intr(VIDEO, &regs);
    movedata(regs.r_es, regs.r_bp, _DS, (unsigned) fp_ptr, 14*256);
}

/* GET_VIDEO_INFO: A VGA or an EGA must be installed for this program */
/* to work. The monitor must be an Enhanced Color or Monochrome */
/* display and the correct adaptor must be active. */
int get_video_info()
{
    union REGS regs;
    unsigned char e_byte;

    /* First check for the presence of an EGA */
    regs.h.ah = 0x12; /* EGA BIOS alternate select */
    regs.h.bl = 0x10; /* return EGA information. */
    int86(VIDEO, &regs, &regs);
    if (regs.h.bl == 0x10) error(1); /* EGA not found */

    /* EGA is present -- is it active? */
    e_byte = peekb(0, 0x487); /* EGA info. byte */
    if (e_byte & 8) error(2); /* EGA not active */

    /* Does the present, active EGA drive a color or mono monitor? */
    if (regs.h.bh) ega_color = FALSE; /* EGA drives a mono monitor */
    else ega_color = TRUE; /* EGA drives a color monitor */

    /* See if EGA drives an Enhanced Color Display */
    if (ega_color) if (!(regs.h.cl == 3 || regs.h.cl == 9)) error(1);
    return (1);
}

/* ERROR: A simple error handler. */
void error(int errnum)
{
    switch (errnum) {
        case 1: printf("\nAn EGA and Enhanced Color or Monochrome Display");
                printf("\nmust be present to use this program.");
                break;

        case 2: printf("\nPlease make the EGA the active adapter");
                printf("\nin order to run this program.");
                break;

        case 3: printf("\nError deallocating program memory.");
                break;

        case 4: printf("\nError deallocating program PSP.");
                break;

        default: break;
    }
    printf("\nProgram exiting.\n");
    exit(0xf); /* Return code for DOS errorlevel */
}

```

End Listings

Moving?

Mail entire ad to:

DR. DOBB'S JOURNAL
P.O. Box 3713
Escondido, CA 92025

Affix your
present label
here.

Please allow 6-8 weeks
for address change
to take effect.

New address _____

City _____ State _____ Zip _____

"How to protect your software by letting people copy it"

By Dick Erett, President of Software Security



Inventor and entrepreneur, Dick Erett, explains his company's view on the protection of intellectual property.

"A crucial point that even sophisticated software development companies and the trade press seem to be missing or ignoring is this:

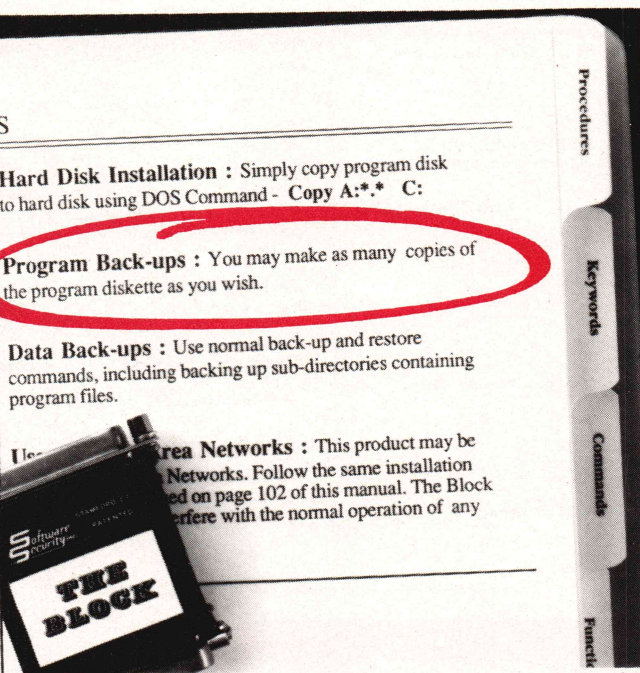
Software protection must be understood to be a distinctively different concept from that commonly referred to as copy protection.

Fundamentally, software protection involves devising a method that prevents unauthorized use of a program, without restricting a legitimate user from making any number of additional copies or preventing program operation via hard disk or LANs.

Logic dictates that magnetic media can no more protect itself from misuse than a padlock can lock itself.

Software protection must reside outside the actual storage media. The technique can then be made as tamper proof as deemed necessary. If one is clever enough, patent law can be brought to bear on the method.

Software protection is at a crossroads and the choices are clear. You can give product away to a segment



Soon all software installation procedures will be as straightforward as this. The only difference will be whether you include the option to steal your product or not.

of the market, or take a stand against the theft of your intellectual property.

"...giving your software away is fine..."

We strongly believe that giving your software away is fine, if you make the decision to do so. However, if the public's sense of ethics is determining company policy, then you are no longer in control.

We have patented a device that protects your software while allowing unlimited archival copies and uninhibited use of hard disks and LANs. The name of this product is The BLOCK™.

The BLOCK is the only patented method we know of to protect your investment. It answers all the complaints of reasonable people concerning software protection.

In reality, the only people who could object are those who would like the option of stealing your company's product.

"...eliminating the rationale for copy-busting..."

Since The BLOCK allows a user to make unlimited archival copies the rationale for copy-busting programs is eliminated.

The BLOCK is fully protected by federal patent law rather than the less effective copyright statutes. The law clearly prohibits the production of work-alike devices to replace The BLOCK.

The BLOCK attaches to any communications port of virtually any microcomputer. It comes with a unique customer product number programmed into the circuit.

The BLOCK is transparent to any device attached to the port. Once it is in place users are essentially unaware of its presence. The BLOCK may be daisy-chained to provide security for more than one software package.

Each software developer devises their own procedure for accessing The BLOCK to confirm a legitimate user. If it is not present, then the program can take appropriate action.

"...possibilities... limited only by your imagination..."

The elegance of The BLOCK lies in its simplicity. Once you understand the principle of The BLOCK, hundreds of possibilities will manifest themselves, limited only by your imagination.

Your efforts, investments and intellectual property belong to you, and you have an obligation to protect them. Let us help you safeguard what's rightfully yours. Call today for our brochure, or a demo unit."

Software Security inc.

870 High Ridge Road Stamford, Connecticut 06905
203 329 8870

BINARY TREES

Listing One (Text begins on page 42.)

```

/* 001 23-Apr-87  tbtree.h

Header file for threaded binary tree functions.

This code is hereby placed into the public domain.

*/

/* define TREE_NODE type */
typedef struct _tree_ent {
    char *word;           /* word ptr for this node */
    int usage;            /* usage counter for word */
    int flags;            /* word flags */
    struct _tree_ent *lchild; /* thread or left child ptr */
    struct _tree_ent *rchild; /* thread or right child ptr */
} TREE_NODE;

/* define bit values for node flags */

#define RBIT (1)          /* right child if set, thread if 0 */
#define LBIT (2)          /* left child if set, thread if 0 */

```

End Listing One

Listing Two

```

/* 001 24-Apr-87  tbtree.c

Functions to create and traverse a threaded binary tree.

James Mathews
Blue Sky Software
172 Manor Drive
Absecon, NJ 08201

This code is hereby placed into the public domain.

*/

```

```

#include <stdio.h>
#include "tbtree.h"

#ifdef LINT ARGS          /* setup for Microsoft C V 4.0 */
#include <malloc.h>
TREE_NODE *talloc();
void add2tree(char *);
TREE_NODE *linsert(TREE_NODE *);
TREE_NODE *rinsert(TREE_NODE *);
TREE_NODE *inorder_succ(TREE_NODE *);
TREE_NODE *inorder_pred(TREE_NODE *);
#else
char *malloc();
void add2tree();
TREE_NODE *talloc(), *linsert(), *rinsert();
TREE_NODE *inorder_succ(), *inorder_pred();
#endif

/* define the root node for the threaded tree */
TREE_NODE root = { "", 0, RBIT, &root, &root };

/*****
A D D 2 T R E E
*****/

void
add2tree(word)           /* add given word to the B-tree */
char *word;
{
    register int cmp;
    register TREE_NODE *tp = &root;

    /* search tree to find word, add it if not there */

    while ((cmp = strcmp(word, tp->word)) != 0) {
        if (cmp < 0)          /* not equal, look to the left? */
            if (tp->flags & LBIT) /* is there a left child? */
                tp = tp->lchild; /* yes, go look at it */

```

(continued on page 60)

The Custom 386 Programmer's Workstation

Looking for a lightning-quick 386 system that's tailored to your needs? CAE/SAR Systems, Inc. will custom-fit you a 386 system more powerful than most on the market. Whether it's a system designed for your program development, artificial intelligence, CAE, or systems design work, CAE/SAR delivers reliable, powerful 386 workstations built for today's programmers.

Based on a proven 386 motherboard, CAE/SAR 386 systems come in dozens of different configurations for memory, disks, floating point and graphics. You can select high speed drives (16 ms), 70Mb, 140Mb, or 300Mb; EGA or mono monitors and cards; and 2.5Mb, 4.5Mb, or 8.5Mb 32-bit RAM— plus other options!

The CAE/SAR 386 systems run Unix and DOS concurrently, and also run OS/2

and Xenix. Floating point options are available for the Intel 387 chip.

Basic Unix/Xenix systems start at \$3,495.

Get a system that fits you perfectly. Call CAE/SAR Systems today for more information.

CAE/SAR Systems, Inc.

P.O. Box 50243
Palo Alto, CA 94303
(415) 949-3816

"The winner, though, was the CAE/SAR 386. Its ESDI hard disk interface made it the fastest of all the machines in the disk access test."

PC Magazine
Dec. 22, 1987

CIRCLE NO. 132 ON READER SERVICE CARD

New Prices

OS/2

WINDOWS FOR DATA®

MULTI-LEVEL
MENU SYSTEM

NESTED
POP-UP FORMS

SCROLLABLE
REGION

CHOICE LIST

CLOCK

POP-UP
WINDOW

RUNNING
TOTALS

MESSAGE
WINDOW

INVOICES: Create Review Print Exit

INVOICE

Invoice No.: 008784 Date: 12/03/87 Time: 16:43:15

Customer:

William Jones
Innovative Software
351 Bulletin Avenue
Needham, MA 02194
(617) 394-5512

Search for customer record? (Y/N): N
Enter customer information? (Y/N): N
Enter billing address? (Y/N): N
Enter marketing information? (Y/N): N

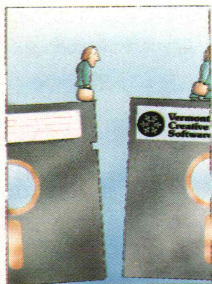
No.	PRODUCT	DESCRIPTION	QUANTITY	PRICE	AMOUNT
5	WDMS	Windows for Data Microsoft	10	295.00	2950.00
6	WDLA	Windows for Data Lattice	5	295.00	1475.00
7	WDTC	Windows for Data Turbo C	5	295.00	1475.00
8	WDXE	Windows for Data XENIX	2	795.00	1590.00
9			0	0.00	0.00

Subtotal: 11325.00
Shipping: 0.00
TOTAL: 11325.00
Payment: 0.00

Cursor keys scroll, ENTER selects and ESC exits choice menu

If you program in C, take a few moments to learn how Windows for Data can help you build a state-of-the-art user interface.

- ✓ Create and manage menus, data-entry forms, context-sensitive help, and text displays — all within windows.
- ✓ Develop window-based OS/2 programs right now, without the headaches of learning OS/2 screen management. Run the same source code in PC DOS and OS/2 protected mode.
- ✓ Build a better front end for any DBMS that has a C-language interface (most popular ones do).



NO WALLS

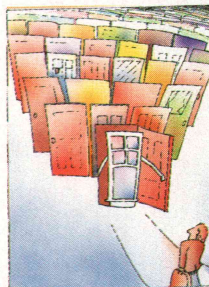
If you've been frustrated by the limitations of other screen utilities, don't be discouraged. You won't run into walls with Windows for Data. Our customers repeatedly tell us how they've used our system in ways we never imagined — but which we anticipated by designing Windows for Data for unprecedented adaptability. You will be amazed at what you can do with Windows for Data.

FROM END TO BEGINNING

Windows for Data begins where other screen packages end, with special features like nested pop-up forms and menus, field entry from lists of choices, scrollable regions for the entry of variable numbers of line items, and an exclusive built-in debugging system.

YOU ARE ALWAYS IN CHARGE

Control functions that you write and attach to fields and/or keys can read, compare, validate, and change the data values in all fields of the form. Upon entry or exit from any field, control functions can call up subsidiary forms and menus, change the active field, exit or abort the form, perform almost any task you can imagine.



OUR WINDOWS WILL OPEN DOORS

Our windows will open doors to new markets for your software. High-performance, source-code-compatible versions of Windows for Data are now available for PC DOS, OS/2, XENIX, UNIX, and VMS. PC DOS

versions are fully compatible with Microsoft Windows. **No royalties.**

MONEY BACK GUARANTEE

You owe it to yourself and your programs to try Windows for Data. If not satisfied, you can return it for a full refund.

Prices: PC DOS \$295, Source \$295. OS/2 \$495. XENIX \$795. UNIX, VMS, please call.

Call: (802) 848-7731

Telex: 510-601-4160 VC SOFT

ext. 31

FAX 802-848-3502



**Vermont
Creative
Software**

21 Elm Ave.
Richford,
VT 05476

BINARY TREES

Listing Two (Listing continued, text begins on page 42.)

```

else {
    tp = linsert(tp); /* no left child, make one */
    break;
}

else /* not left, look right */

if (tp->flags & RBIT) /* is there a right child? */
    tp = tp->rchild; /* yes, look it over */
else {
    tp = rinsert(tp); /* no right child, make one */
    break;
}

if (cmp == 0) /* did we find the word? */
    tp->usage++; /* if so bump usage count */

else { /* must have added new word */

    tp->word = word; /* point it to the word */
    tp->usage = 1; /* new word in town */

}

/*****
R I N S E R T
*****/

static TREE_NODE *
rinsert(tp) /* insert new node as right child of tp */
register TREE_NODE *tp;
{
    register TREE_NODE *new;

    if (new = malloc()) { /* alloc new entry */

        new->rchild = tp->rchild; /* new gets what tp had */
        new->flags = tp->flags & RBIT; /* as a right child */

        new->lchild = tp; /* lchild is thread to tp */
    }
}

```

```

tp->rchild = new; /* new is rchild of tp */
tp->flags |= RBIT; /* real node, not thread */

return(new);

/*****
L I N S E R T
*****/

static TREE_NODE *
linsert(tp) /* insert new node as left child of tp */
register TREE_NODE *tp;
{
    register TREE_NODE *new;

    if (new = malloc()) { /* alloc new entry */

        new->lchild = tp->lchild; /* new gets what tp had */
        new->flags = tp->flags & LBIT; /* as a left child */

        new->rchild = tp; /* rchild is thread to tp */

        tp->lchild = new; /* new is lchild of tp */
        tp->flags |= LBIT; /* real node, not thread */

    }

    return(new);

/*****
T A L L O C
*****/

static TREE_NODE *
malloc() { /* allocate a TREE_NODE */

#define NODES_PER_ALLOC (50)

```

(continued on page 63)

Troff Support for Laser Printers!

EROFF™ now supports the HP LaserJet, Imagen, Laserwriter, and all POSTSCRIPT® printers!

CRT screen previewers for rough drafting are now included with **EROFF™**.

Full graphics previewers for SUN and X-Windows are also available.

Our enhanced **troff** allows inclusion of graphics directly into your documents. Available on MS-DOS and 36 different UNIX systems.

IMAGES



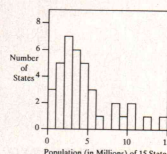
EQUATIONS

$$A'' = \lim \sum f(x_k) \Delta x$$

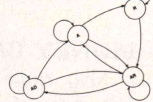
$$\frac{\partial^2 \phi}{\partial x^2} = \frac{\partial^2 \phi}{\partial x \partial y}$$

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

GRAPHS/PLOTS



DIAGRAMS



TABLES

Food	Composition of Foods		
	Percent by Weight		
	Protein	Fat	Carb. hydrate
Apples	4	5	13.0
Haitout	18.4	5.2	
Lima beans	7.5	8	22.0
Milk	3.3	4.0	8.0
Mushrooms	3.5	4	
Rye bread	9.0	6	52.7

8031

FORTH DEVELOPMENT ENVIRONMENT

Take advantage of Bryte's tools to make your job easier:

- Bryte's development environment uses BRYTE-FORTH on the actual production hardware during product development. No emulators, no changes, no surprises.
- Optional PC-based cross-development tools use DOS files as microcontroller mass storage. These files can be used to generate compact EPROM images, detailed listings, and cross-references.

Why not start developing the Bryte way today?

BRYTE-FORTH 8031 EPROM	100.00
(includes 130 page User's Manual)	
Utility disk(s)	65.00*
Cross-compiler/Cross-assembler	235.00*
8031 unlimited quan. license	1000.00*

* Includes complete source code

bryte computers, inc.

P.O. Box 46
Augusta, ME 04330-0046

207/547-3218



Elan Computer Group, Inc.
410 Cambridge Ave., Suite A, Palo Alto, CA 94306
415.322.2450

CIRCLE NO. 134 ON READER SERVICE CARD

CIRCLE NO. 135 ON READER SERVICE CARD

Special Issue. Special Advertising Opportunity.

Until now there simply hasn't been a single reference source for Macintosh software developers.

Dr. Dobb's Journal changes the picture...
with one information-packed special issue:
Dr. Dobb's Macintosh Special.

Dr. Dobb's Macintosh Special has more than
100 pages of essential information that
Mac programmers will turn to time and time again.

If you've got a product for the Macintosh
software pro, *Dr. Dobb's Macintosh Special*
offers a unique advertising opportunity.

Dr. Dobb's Macintosh Special provides detailed
information exclusively for the Macintosh software
developer, with features like ...

- The Macintosh programming environment. How software professionals will maintain the momentum of 1987.
- The Mac II. Why the Mac II is the programmer's machine of choice in 1988.
- HyperCard. An in-depth technical examination... plus source code and product guides.

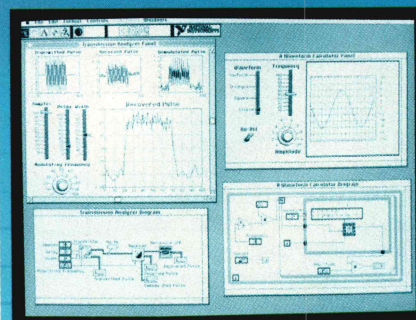
Dr. Dobb's Macintosh Special is a tool that will be
saved and referred to over and over. It will be
strategically placed next to thousands of professional
developers' Macs...and your marketing message will
produce sales for months.

For our cost-effective advertising rates, call today.
Think about the opportunity *Dr. Dobb's Macintosh
Special* represents...but don't think too long.
Space is limited.

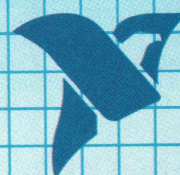
Ferris Ferdon
Director of Advertising
(415) 366-3600

LabVIEW

The Scientific
and Engineering
Software System
for
Data Acquisition
and
Instrument
Control



- Data acquisition, analysis, management, and presentation
- User-designed front panels
- User-designed block diagrams for efficient software construction
- Support for IEEE-488, A/D, D/A, DI/O, DMA, and Timing
- Libraries for digital signal processing, statistics, curve fitting, data acquisition, and instrument control
- Modeling and simulation



NATIONAL INSTRUMENTS™

The Leader in IEEE-488

12109 Technology Blvd.
Austin, Texas • 78727-6204

CALL FOR FREE CATALOG
800/531-4742 • 512/250-9119

CIRCLE NO. 136 ON READER SERVICE CARD

THE BEST OF BOTH WORLDS

Developing an application used to be easy — all you had to do was program it. But today, with countless languages, compilers, libraries, databases, editors, debuggers, and other tools, it is choosing the right development software that creates the real problem.

Now *The Andsor Collection* introduces a unique solution: a collection of sophisticated development tools, which you can use on their own, or together with your old ones.

The Andsor Collection is, of course, the superb application development system that programmers, VARs, and other developers have been using for over two years. And starting with Version 2.2, *The Andsor Collection* has acquired a new dimension: now you can access all its functions from within another program!

Think of it as a comprehensive, universal, language independent library. But *The Andsor Collection* is not a collection of subroutines: it is a seamless, integrated, interactive environment, specifically designed to expedite application development.

Whether you use C, Pascal, Cobol, Fortran, Basic, or any other language, *The Andsor Collection* can enhance your applications dramatically. Whether you add functions to an old program, or you write a new one, you can make them faster, more efficient, and more appealing.

Use *The Andsor Collection* to implement an entire application, or just portions of an application. You can, for example, create a windowed environment, add attractive data entry functions, define indexed data file structures, produce sophisticated reports or forms, and so on.

Although *The Andsor Collection* has far more features than other development systems, it is only one tenth their size. So the entire system can stay in memory, keeping all functions instantly accessible.

And *The Andsor Collection* is famous for its unique interactive environment. There is no conversion or translation — modify a procedure, a file definition or relation, a data entry screen, or anything else, and the change takes effect immediately, even while the application is running! Application development is a new experience.

The application users will benefit too. *The Andsor Collection* is amazingly fast, and since all data is in variable length format, the files take a fraction of the space needed with other systems. So not only will you develop your applications sooner, but they will be more efficient too. Whether you use *The Andsor Collection* alone, or to enhance a program.

So get the best of both worlds. Order *The Andsor Collection* today, and discover a whole new environment, without giving up your old development tools or your existing applications. Moreover, *The Andsor Collection* will be useful with all your future applications and languages.

You won't find a better value in development software: one program that is both a powerful stand-alone application development system, and a unique language independent collection of software tools; plus the run-time interpreter with unlimited royalty-free distribution. All for an incredibly low price. And with our 60 day money back guarantee*, you have little to lose and a lot to gain.

System Features

The Andsor Collection is the most versatile application development environment. And when using it with your programs, all its countless features can become part of your application. Hundreds of commands, functions, and options, are available to help you implement any application.

No list can be complete, so here are just some of these features: powerful database functions, maintenance-free multi-index data files, variable length data fields, unlimited file relations, complete window management, unique text processing functions, flexible data tables, powerful inquiry and reporting functions, versatile data entry capabilities, flexible procedural language, automatic error handling, extensive computational capabilities, data analysis and statistics, unique programmable charts, many printing functions, data communications, convenient system log file, full control over color attributes. And much, much more.

How It Works

This is simple and ingenious. Your program and *The Andsor Collection* reside in memory together (you load *The Andsor Collection*, which then loads your program). To transfer control to those portions of the application implemented in *The Andsor Collection*, you simply issue a software interrupt in your program, exactly the way DOS and BIOS functions are called. (If you are not familiar with this, examples in the manual show you how to do it.)

While in *The Andsor Collection*, the operation is identical to its operation when used alone. Finally, one command returns control to your program. And if you need this, simple commands transfer data directly to and from memory areas in your program (a number of formats are possible, all in standard ASCII, compatible with any language). Both *The Andsor Collection* and your program run as ordinary DOS programs. And you can also use them with other software (such as permanently resident programs).

"We looked at several systems and decided to standardize on *The Andsor Collection*. We are using it in many applications in several departments."

Gordon Evers, Director of Information Services, Public Utilities Commission, Brantford, Ontario

"With *The Andsor Collection* we have achieved faster development and more efficient applications, which is important in large and complex projects like our Court Management System."

Dr. Mark Schrager, Consultant, Municipal Computer Services, Rochester, New York

"*The Andsor Collection* is unequalled when we need a solution in a hurry. Applications that we have implemented include modeling, data acquisition and analysis, and reporting."

Joe Blask, Engineering Support, General Motors, Troy, Michigan

ANDSOR®

ANDSOR RESEARCH INC.
390 Bay Street, Suite 2000
Toronto, Ontario M5H 2Y2
(416) 245-8073

To order call toll free
(U.S. and Canada)

1-800-628-2828
Ext. 535

The Andsor Collection™

\$145 (includes shipping*)

Visa, MC, AmEx, Check

*Price includes shipping in the U.S. and Canada. Please add \$10 for shipping to other countries. If you return the software, \$8 will be deducted from the refund, to cover our shipping cost.

System requirements: any IBM PC or PS/2 or fully compatible, 250K+ (excluding DOS and other programs), one disk drive or hard disk, monochrome or color monitor, DOS 2.0+ or OS/2

© 1988 Andsor Research Inc. Andsor is a registered trademark and *The Andsor Collection* is a trademark of Andsor Research Inc. IBM is a registered trademark and IBM PC, PS/2, OS/2 are trademarks of IBM Corporation



Listing Two (Listing continued, text begins on page 42.)

```
static TREE_NODE *tp;
static int tidix = NODES_PER_ALLOC;

/* this routine allocates space for TREE_NODE nodes. It
exists to cut down on the number of calls to malloc(). */

if (tidix < NODES_PER_ALLOC) /* free nodes from last alloc? */
    return(&tp[tidix++]); /* yes, return the next one */

/* allocate another block of TREE_NODE nodes */
tp = (TREE_NODE *) malloc(sizeof(TREE_NODE) * NODES_PER_ALLOC);

if (tp == NULL) {
    printf("\nOut of memory in talloc()\n");
    exit();
}

tidix = 1; /* return # 1 next time */
return(tp); /* return # 0 this time */

/*****
IN ORDER _ SUCC
*****/

TREE_NODE *
inorder_succ(tp) /* return in-order successor of tp */
register TREE_NODE *tp;
{
    register TREE_NODE *next;

    /* if the right child is a thread, it's the successor node */

    next = tp->rchild;

    /* but if the right child is a real node, the successor is
left-most child of the right child */

    if (tp->flags & RBIT) /* real right child? */
        while (next->flags & LBIT) /* find left-most */
            next = next->lchild; /* child of that */

    return(next); /* return successor */
}

/*****
IN ORDER _ PRED
*****/

TREE_NODE *
inorder_pred(tp) /* return in-order predecessor of tp */
register TREE_NODE *tp;
{
    register TREE_NODE *prev;

    /* if the left child is a thread, it's the predecessor */

    prev = tp->lchild;

    /* but if the left child is a real node, the predecessor is
right-most child of the left child */

    if (tp->flags & LBIT) /* real left child? */
        while (prev->flags & RBIT) /* find right-most */
            prev = prev->rchild; /* child of that */

    return(prev); /* return predecessor */
}
```

End Listings



Now You Have A Choice

in Software Protection

CHOICE #1

The Secom Key

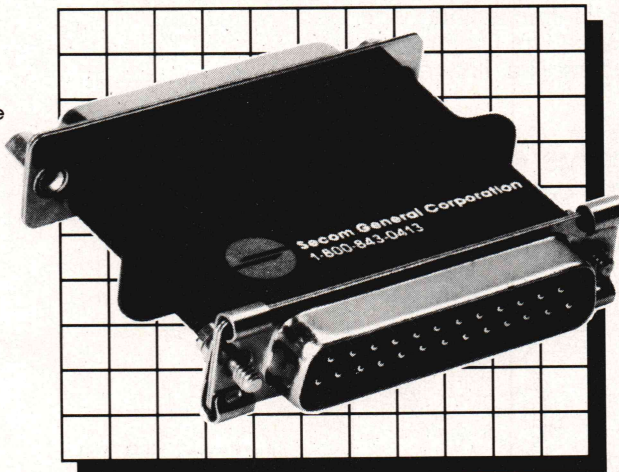
The Secom Key provides effective software protection while insuring customer satisfaction. It eliminates the problems normally associated with copy protection. The Secom Key is designed for software packages which are reproduced identically. Available in quantities, for as low as \$21.95.

Both the Secom Key AND The Memory Key:

- ☐ are completely transparent to end user
- ☐ will not interfere with peripheral operations
- ☐ don't occupy disk drives
- ☐ allow unlimited backup copies
- ☐ are easily installed
- ☐ are very small in size



Secom General Corporation
1829 E. Franklin Street #500
Chapel Hill, North Carolina 27514
(919) 942-8500



Secom offers alternatives!

If you would like a demonstration package or additional information, please write or call:

1-800-843-0413

CHOICE #2

The Memory Key

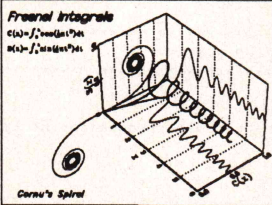
The Memory Key offers special flexibility. It comes with unique software which permits the use of its available read/write memory. Each byte of memory can be addressed individually or in groups for specific identification. Also, numerical information can be transferred between the Memory Key and your software and acted upon. The Memory Key comes with special error checking abilities providing 100% reliability. Example applications are:

- ☐ modular package control
- ☐ serialization
- ☐ software customization
- ☐ demo control
- ☐ auditable and easy software leasing
- ☐ any "counter" operation

The ease of use, cost effectiveness and functionality of the Memory Key allows for previously unavailable controls and applications.

GraphiC™

**Publication
quality graphics
on your IBM® PC**



- linear, log, polar plots
- bar charts, Smith charts
- 3D curves, 3D surfaces
- 6 curve types, 8 markers
- thick lines, panel fills
- 15 fonts, font editor
- 4096 x 3120 resolution
- zoom, pan, window plots
- high resolution printer & plotter dumps in color

**Over 150 C and assembler
routines for full control**

\$395 with source code.

For personal use only.

MOST HARDWARE IS SUPPORTED

Scientific Endeavors Corporation

Route 4, Box 79 Kingstons, TN 37763 (615) 376-4146

CIRCLE NO. 139 ON READER SERVICE CARD

VTEK™

**DEC® VT100/VT52
and Tektronix®
4010, 4014, & 4105
Terminal Emulator**

- 20 user-defined keys
- large scroll back buffer
- hardware 132 columns
- Kermit and XMODEM
- up to 800x600 screen resolution on EGAs
- zoom, pan, window plots
- "hot key" to DOS
- all VT100 keys, long and short breaks
- ANSI extensions to VT100 for multi-color text
- scrolling VT100 window on graphics screen
- convert files to .GEM & .PIC formats

**\$150. Site and source
code licenses available**

B-EDIT™

**Our new binary editor
for programmers - \$29**

TO THE MACS

Listing One (Text begins on page 90.)

```
/* resource decompilation of custom controls PROJ.Rsrc */
/* decompilation performed by Alan Dahlbom's ResTools 2.01 */
/* PICTs, ICONs, ICN#s, and CDEFs not included */
```

```
resource 'BNDL' (128)
{
    'CuCo', 0,
    {
        'FREF', (0, 128);
        'ICN#', (0, 128)
    }
};
```

```
resource 'CNTL' (1)
{
    {0, 0, 28, 130},
    0,
    visible,
    0,
    0,
    640,
    0x0,
    "Press Me To Quit"
};
```

```
resource 'CNTL' (2)
{
    {0, 0, 36, 145},
    6,
    visible,
    18,
    0,
    643,
    0x140000,
    "War Is Peace"
};
```

```
resource 'CNTL' (3)
{
    {0, 0, 40, 40},
    0,
    visible,
    0,
    0,
    652,
    0x1e,
    ""
};
```

```
resource 'CNTL' (4)
{
    {0, 0, 30, 178},
    2,
    visible,
    14,
    768,
    641,
    0x280000,
    "Evolve Or Dissolve"
};
```

```
resource 'CNTL' (5)
{
    {240, 300, 297, 361},
    0,
    visible,
    0,
    0,
    645,
    0x330032,
    "\0x00"
};
```

```
resource 'CNTL' (6)
{
    {0, 0, 78, 80},
    0,
    visible,
    0,
    0,
    647,
    0x3d003c,
    ""
};
```

```
resource 'CNTL' (7)
{
    {0, 0, 50, 45},
    0,
    visible,
    0,
    0,
    646,
    0x46,
    ""
};
```

```
resource 'CNTL' (8)
{
    {0, 0, 200, 116},
    0,
    visible,
    0,
    0,
    644,
    0x50,
    ""
};
```

```
resource 'CNTL' (9)
{
    {0, 12, 50, 100},
    0,
    visible,
    0,
    0,
    648,
    0x5a,
    ""
};
```

```
resource 'CNTL' (10)
{
    {0, 0, 66, 80},
    0,
    visible,
    0,
    0,
    649,
    0x650064,
    ""
};
```

```
resource 'CNTL' (11)
{
    {0, 0, 36, 36},
    0,
    visible,
    0,
    0,
    650,
    0x6e,
    ""
};
```

```
resource 'CNTL' (12)
{
    {0, 0, 32, 32},
    0,
    visible,
    0,
    0,
    651,
    0x790078,
    ""
};
```

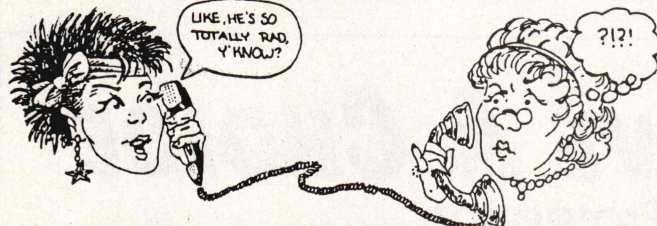
```
resource 'CNTL' (13)
{
    {0, 0, 40, 38},
    0,
    visible,
    0,
    0,
    652,
    0x82,
    "\0x00"
};
```

```
resource 'CNTL' (14)
{
    {240, 300, 277, 337},
    0,
    visible,
    0,
    0,
    653,
    0x8d008c,
    "\0x00"
};
```

```
resource 'CNTL' (15)
{
    {0, 0, 40, 40},
    0,
    visible,
    0,
    0,
    654,
    0x96,
    "\0x00"
};
```

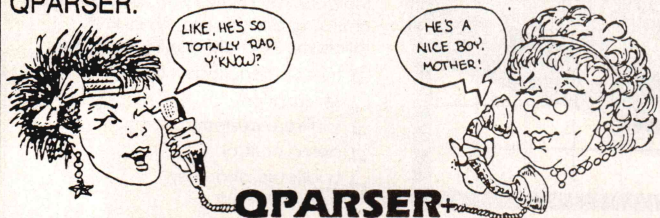
```
resource 'CNTL' (16)
{
    {0, 0, 40, 40},
    0,
    visible,
    0,
    0,
    655,
    0xa100a0,
    "\0x00"
};
```

```
resource 'CNTL' (17)
{
    {0, 0, 187, 85},
    0,
    visible,
    0,
    0,
    645,
    0xab00aa,
    ""
};
```



Now you can use **QPARSER+** to develop compilers, interpreters, complex user-interfaces, language & file format translators (i.e. Pascal to C, Bit Map to Postscript), language debuggers like lint, etc..

Develop language translators in C and Pascal within the IBM PC/XT/AT or VAX/VMS environments. A new user manual, automated syntax tree construction and an advanced code generation language are just a few of the improvements over the original QPARSER.



QPARSER+

Another translation by QPARSER™

**Limited Time Offer
Just \$300**

— FREE demo disk available

QCAD Systems

1164 Hyde Avenue, San José CA 95129 (408) 727-6884

Outside Calif. call TOLL-FREE (800) 538-9787

CIRCLE NO. 140 ON READER SERVICE CARD


```

";
};
resource 'CNTL' (18)
{
    {0, 0, 18, 120},
    3,
    visible,
    10,
    0,
    640,
    0x0,
    "Turn Off Some Buttons"
};
resource 'CNTL' (19)
{
    {0, 0, 18, 150},
    3,
    visible,
    12,
    0,
    642,
    0x0,
    "Turn On Some Buttons"
};
resource 'CNTL' (20)
{
    {0, 0, 24, 24},
    0,
    visible,
    0,
    0,
    645,
    0xc900c8,
    ""
};
resource 'CNTL' (21)
{
    {0, 0, 12, 90},
    3,
    visible,
    9,
    0,
    640,
    0x0,
    "\0xa91987 Stan Krute"
};
userdefined resource 'CuCo' (0)
{
    pstring: "custom controls demo
version 1.0 (©1987 by Stan Krute
all rights reserved"
};
resource 'DITL' (1)
{
    {238, 326, 266, 456},
    Control {enabled, 1};
    {7, 6, 43, 151},
    Control {enabled, 2};
    {129, 285, 169, 325},
    Control {enabled, 3};
    {174, 197, 204, 375},
    Control {enabled, 4};
    {201, 120, 258, 181},
    Control {enabled, 5};
    {4, 230, 82, 310},
    Control {enabled, 6};
    {12, 169, 62, 214},
    Control {enabled, 7};
    {71, 0, 271, 116},
    Control {enabled, 8};
    {213, 199, 263, 287},
    Control {enabled, 9};
    {69, 141, 135, 221},
    Control {enabled, 10};
    {5, 320, 41, 356},
    Control {enabled, 11};
    {138, 226, 170, 258},
    Control {enabled, 12};
    {90, 238, 130, 276},
    Control {enabled, 13};
    {105, 335, 142, 372},
    Control {enabled, 14};
};

```

```

{47, 330, 87, 370},
Control {enabled, 15};
{148, 142, 188, 182},
Control {enabled, 16};
{22, 377, 209, 462},
Control {enabled, 17};
{49, 18, 67, 138},
Control {enabled, 18};
{214, 305, 232, 455},
Control {enabled, 19};
{101, 297, 125, 321},
Control {enabled, 20};
{5, 364, 17, 454},
Control {enabled, 21}
};
resource 'DITL' (210)
{
    {10, 10, 178, 286},
    Picture {enabled, 210}
};
resource 'DLOG' (1)
{
    {0, 0, 272, 462},
    1,
    invisible,
    noGoAway,
    0x0,
    1,
    "New Dialog"
};
resource 'DLOG' (210)
{
    {0, 0, 188, 296},
    1,
    invisible,
    noGoAway,
    0x0,
    210,
    "New Dialog"
};
resource 'FREF' (128)
{
    'APPL',
    0,
    ""
};
resource 'MENU' (1)
{
    1,
    0,
    0xffffffff,
    enabled,
    "custom controls demo",
    {
};
resource 'STR' (20)
{
    "Peace Is War"
};
resource 'STR' (40)
{
    "Grow Up Or Blow Up"
};
};

```

End Listing One

Listing Two

```

asm rectCDEF.asm Exec QUED/M 2.04
link rectCDEF.link Exec QUED/M 2.04
RMaker rectCDEF.R0 Exec QUED/M 2.04
RMaker rectCDEF.R1 Exec QUED/M 2.04
RMaker rectCDEF.R2 custom controls demo QUED/M 2.04

```

End Listing Two

(Listing Three continued on next page.)

**FORTRAN
TO C**

FOR_C™

Discover the power of C!

Use **FOR_C™** to convert standard FORTRAN-77 (with MIL-STD-1753 and common FORTRAN-77 extensions) into ANSI C with great speed and ease of use.

- 99+% conversion rate on standard FORTRAN-77
- Translate code between PC's and mini-computers
- Utilize a built-in, C-like preprocessor for easy code customization
- **FOR_C™** generates C-style prototypes and checks your FORTRAN source for consistent usage, allowing you to customize function calls in C. This results in the best and most concise C code possible
- Produces readable, maintainable code with precise, and accurate translations
- Includes C SOURCE to all run-time libraries — now a complete solution to your portability problems!

Simply the best FORTRAN to C translator available — at any price!

Order your copy today!

Introductory Offer: \$650⁰⁰ MS-DOS

AFTER MAY 1, 1988: \$750.00

Package includes six months' free support and upgrades

COBALT BLUE

1683 Milroy • Ste 101 • San Jose • CA 95124
(408) 723-0474

CIRCLE NO. 141 ON READER SERVICE CARD

TURBO C QUICK C LET'S C DESMET C DATALIGHT C ECO-C
LATTICE C MICROSOFT C AZTEC C COMPUTER INNOVATIONS C

NEW --- Limited time offer.

Peacock System's CBTREE

Object library for only \$49!

Our FULL COMMERCIAL VERSION of CBTREE in object library format is being offered for the amazingly low price of \$49.

CBTREE provides you with easy to use functions that maintain key indexes on your data records. These indexes provide you with fast, keyed access, using the industry standard B+tree access method.

Everything you need to fully utilize CBTREE in your applications is included. The CBTREE source code can be purchased later at any time for the \$110 difference. Example source programs and utilities are included FREE.

CBTREE source library	\$159
Object library only	\$49

This limited time offer is simply too good to refuse. Peacock's standard ROYALTY FREE, UNCONDITIONAL MONEY-BACK GUARANTEE, AND FREE TECHNICAL SUPPORT applies to this offer.

To order or for additional information
call 1-800-346-8038 or (703) 847-1743 or write:



PEACOCK SYSTEMS, INC.

PEACOCK SYSTEMS, INC.
2108 GALLOWES ROAD, SUITE C
VIENNA, VA 22180

Trademarks: Turbo C (Borland); Quick C (Microsoft); Let's C (Mark Williams); DeSmet C (DeSmet Software); Datalight (Datalight); Lattice C (Lattice); Microsoft C (Microsoft); Aztec C (Manx Software); Computer Innovations C (Computer Innovations); Eco-C (Ecosoft, Inc.).

CIRCLE NO. 142 ON READER SERVICE CARD

NROFF/PC™

The REAL Thing for DOS

NROFF/PC is a complete text formatting system for MS-DOS systems. Including:

NROFF The *powerful* UNIX text formatter

TBL A tool to assist with the layout of tabular material in *Nroff* documents

MM A comprehensive *Nroff* macro package for preparing books and technical manuals

NEQN A tool for describing mathematical equations in *Nroff* documents

- All tools are a complete port from the AT&T Documentor's Workbench 2.0
- It's **Fast!** We've modified *Nroff* especially for DOS for lightning speed
- Supports any Dot Matrix printer and many laser printers
- Specially Priced At **\$99**
- A complete *Troff* typesetting system is available **NOW** for LaserJet and PostScript printers on MS-DOS for \$695, XENIX and Microport UNIX for \$795.



Elan Computer Group, Inc.
410 Cambridge Ave., Suite A
Palo Alto, CA 94306
(415) 322-2450

Visa and MasterCard Accepted

Unix is a trademark of AT&T
MS-DOS and Xenix are trademarks of Microsoft

TO THE MACS

Listing Three (Text begins on page 90.)

```
*----- file information -----*
*
* rectCDEF.Asm
*
*
* Assembly language source code for several styles of button controls
* The assembled code is meant to be a CDEF resource
*
* The CDEF provides 16 styles of rectangular button controls
* A button can: (1) contain text, an ICON, or a PICT
*               (2) can be outlined, or shadow-outlined
*               (3) if it contains an ICON or a PICT, can be bare
*               (4) indicate pressing via inversion or a content change
*
* Edited with QUED/M 2.04
* Compiled under MDS 2.01
*
* Written and ©1987 by Stan Krute. All rights reserved. No part of this
* file, or the object code it leads to, may be reproduced, in any form or
* by any means, without the express written permission of the author and
* copyright holder.
*
* Timestamp:      6:51 pm PST                      November 16, 1987
* Spacestamp:    18617 Camp Creek Road Hornbrook, California  96044
*
* This file looks good in 9 point Courier, QUED/M 2.04 tabs set to 3
*-----*
```

```
*----- using the CDEF -----*
```

```
; details on using the CDEF resource produced by this code
```

```
; a custom control is most easily specified via a CNTL resource,
; which uses a procID to indicate the CDEF resource ID and variation code
```

```
; for this CDEF, the resource ID is 40
```

```
; there are 16 variations, as follows:
```

variation	content	border	highlighting via	proc ID
0	text	outlined	inversion	640
1	text	outlined	content change	641
2	text	shadowed	inversion	642
3	text	shadowed	content change	643
4	PICT	bare	inversion	644
5	PICT	bare	content change	645
6	PICT	outlined	inversion	646
7	PICT	outlined	content change	647
8	PICT	shadowed	inversion	648
9	PICT	shadowed	content change	649
10	ICON	bare	inversion	650
11	ICON	bare	content change	651
12	ICON	outlined	inversion	652
13	ICON	outlined	content change	653
14	ICON	shadowed	inversion	654
15	ICON	shadowed	content change	655

```
; for TEXT variations, select a font via the CNTL's ctrlValue field:
; -1 for the window's font, 0 for the System font, 1 for the application font,
; anything else a standard Mac font number
; in the case of a font other than the System or window's font, store the font
; style in the high byte of the ctrlMin field, and the font size in the
; low byte of the ctrlMax field
```

```
; for PICT and ICON variations, store a resource ID indicating the PICT or ICON
; resource in the low word of the CNTL's refCon field
```

```
; for variations that indicate highlighting via a content change, store a resource
; ID indicating the highlighted-state STR (for text variations), PICT, or
; ICON resource in the high word of the CNTL's refCon field
```

```
; when figuring the size of a text button's boundsRect :
; be sure to size the button so the text will fit. Successive approximation works
; well. Rule of thumb for an initial height: 8 greater than the font size.
```

```
; when figuring the size of a PICTURE button's CNTL's boundsRect :
; if the picture has no outline, try a boundsRect that matches the PICT's boundsRect
; if the picture is outlined, try a boundsRect that's at least 4 wider and 4
; higher -- that size lets the picture perfectly match the button's interior
; pictures in larger boundsRects get centered
```

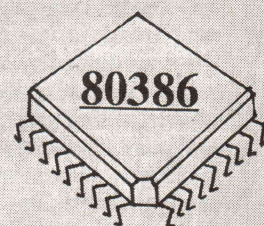
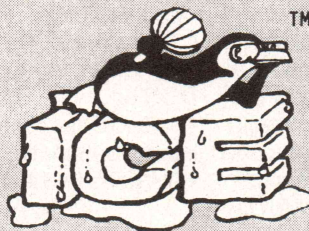
```
; when figuring the size of a ICON button's CNTL's boundsRect :
; if the icon has no outline, try a boundsRect that matches the ICON's boundsRect
; if the icon is outlined, try a boundsRect that's at least 4 wider and 4
; higher -- that size lets the icon perfectly match the button's interior
; icons in larger boundsRects get centered
```

(continued on page 69)

SERIOUS DEBUGGING AT A REASONABLE PRICE



Soft -



All the speed and power of a hardware-assisted debugger at a software price - \$386

Features

Real-time break points on:

- Memory locations
- Memory ranges
- Execution
- I/O ports
- Interrupts (hardware and software)

Dual monitor support

System memory map

Regain control with a keystroke

Even with the following code:

```
CLI
MOV AL,0FFH
OUT 21H,AL
JMP $
```

and much, much more

How Soft-ICE works

Soft-ICE unleashes the power of the 80386 to surround your program in a virtual machine. This gives you complete control of the DOS environment. Soft-ICE uses 80386 protected mode features, such as paging, I/O privilege level, and break point registers, to add real-time hardware-level breakpoints to your existing DOS debugger. To use Soft-ICE you simply pop the Soft-ICE window up through a key sequence, set your hard break points, then return to your soft debugger. As the target program is executing, Soft-ICE recognizes when the breakpoint conditions have been reached and gives control back to your soft debugger. And this is all done at full 80386 speed! Soft-ICE can also be used in stand-alone mode. This comes in handy if you are debugging loadable device drivers, interrupt handlers, or terminate and stay resident programs. All of the standard debugging features are available to help you find the most difficult systems problems.

Benefits

- **Works with CodeView** -- To get you up and going as fast as possible, Soft-ICE is designed to work with your existing software debuggers such as CodeView and Periscope I & II.
- **Breaks the 640K barrier** -- If you have extended memory, Soft-ICE takes up ZERO bytes of memory in the first 1MB of address space. This means you can load and debug your largest programs.
- **Power of an in-circuit emulator** -- At only \$386, you can give every member of your software development team the power of an in-circuit emulator.
- **AT compatible 80386 PC's** -- Soft-ICE works with all AT compatible 80386 PC's, such as COMPAQ's Deskpro 386 and the IBM Model 80.
- **Easy to learn** -- Soft-ICE is so easy to learn that you can be finding bugs with Soft-ICE by the time you could install a hardware-assisted debugger.

"Since Soft-ICE doesn't take up any of my memory I have it in my AUTOEXEC.BAT to load every day... It has saved me at least one month's time on my latest device driver program." Peter Ricker, President of Maverick Software

30 day money-back satisfaction guarantee. Visa and Master Card accepted. Ask about our coupon program.
To order or to get more information, call (603) 888-2386

NU-MEGA TECHNOLOGIES

P.O. BOX 7607
NASHUA, NH 03060-7607

CIRCLE NO. 144 ON READER SERVICE CARD

UNLEASH YOUR 80386!

Your 80386-based PC should run two to three times as fast as your old AT. This speed-up is primarily due to the doubling of the clock speed from 8 to 16 MHz. The new MicroWay products discussed below take advantage of the real power of your 80386, which is actually 4 to 16 times that of the old AT! These new products take advantage of the 32 bit registers and data bus of the 80386 and the Weitek 1167 numeric coprocessor chip set. They include a family of MicroWay

80386 compilers that run in protected mode and numeric coprocessor cards that utilize the Weitek technology.

The benefits of our new technologies include:

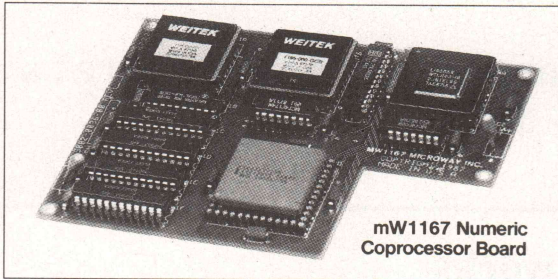
- An increase in addressable memory from 640K to 4 gigabytes using MS-DOS or Unix.
- A 12 fold increase in the speed of 32 bit integer arithmetic.
- A 4 to 16 fold increase in floating point

speed over the 80387/80287 numeric coprocessors.

Equally important, whichever MicroWay product you choose, you can be assured of the same excellent pre- and post-sales support that has made MicroWay the world leader in PC numerics and high performance PC upgrades. For more information, please call the Technical Support Department at

617-746-7341

After July 1988 call 508-746-7341



mW1167 Numeric Coprocessor Board

MicroWay® 80386 Support

MicroWay 80386 Compilers

NDP Fortran-386 and **NDP C-386** are globally optimizing 80386 native code compilers that support a number of Numeric Data Processors, including the 80287, 80387 and mW1167. They generate mainframe quality optimized code and are syntactically and operationally compatible to the Berkeley 4.2 Unix f77 and PCC compilers. MS-DOS specific extensions have been added where necessary to make it easy to port programs written with Microsoft C or Fortran and R/M Fortran.

The compilers are presently available in two formats: Microport Unix 5.3 or MS-DOS as extended by the Phar Lap Tools. MicroWay will port them to other 80386 operating systems such as OS/2 as the need arises and as 80386 versions become available.

The key to addressing more than 640 kbytes is the use of 32-bit integers to address arrays. NDP Fortran-386 generates 32-bit code which executes 3 to 8 times faster than the current generation of 16-bit compilers. There are three elements each of which contributes a factor of 2 to this speed increase: very efficient use of 80386 registers to store 32-bit entities, the use of inline 32-bit arithmetic instead of library calls, and a doubling in the effective utilization of the system data bus.

An example of the benefit of excellent code is a 32-bit matrix multiply. In this benchmark an NDP Fortran-386 program is run against the same program compiled with a 16-bit Fortran. Both programs were run on the same 80386 system. However, the 32-bit code ran 7.5 times faster than the 16-bit code, and 58.5 times faster than the 16-bit code executing on an IBM PC.

NDP FORTRAN-386™\$595
NDP C-386™\$595

MicroWay Numerics

The **mW1167™** is a MicroWay designed high speed numeric coprocessor that works with the 80386. It plugs into a 121 pin "Weitek" socket that is actually a super set of the 80387. This socket is available on a number of motherboards and accelerators including the AT&T 6386, Tandy 4000, Compaq 386/20, Hewlett Packard RS/20 and MicroWay Number Smasher 386. It combines the 64-bit Weitek 1163/64 floating point multiplier/adder with a Weitek/Intel designed "glue chip". The mW1167™ runs at 3.6 MegaWhetstones (compiled with NDP Fortran-386) which is a factor of 16 faster than an AT and 2 to 4 times faster than an 80387.

mW1167 16 MHz\$1495

mW1167 20 MHz\$1995

Monoputer™ - The INMOS T800-20 Transputer is a 32-bit computer on a chip that features a built-in floating point coprocessor. The T800 can be used to build arbitrarily large parallel processing machines. The Monoputer comes with either the 20 MHz T800 or the T414 (a T800 without the NDP) and includes 2 megabytes of processor memory. Transputer language support from MicroWay includes Occam, C, Fortran, Pascal and Prolog.

Monoputer T414-20 with 2 meg¹ ...\$1495

Monoputer T800-20 with 2 meg¹ ...\$1995

Quadputer™ can be purchased with 2, 3 or 4 transputers each of which has 1 or 4 megabytes of memory. Quadputers can be cabled together to build arbitrarily fast parallel processing systems that are as fast or faster than today's mainframes. A single T800 is as fast as an 80386/mW1167 combination!

Biputer™ T800/T414 with 2 meg¹\$3495

Quadputer 4 T414 with 4 meg¹ ...\$6000

¹Includes Occam

80386 Multi-User Solutions

AT8™ - This intelligent serial controller series is designed to handle 4 to 16 users in a Xenix or Unix environment with as little as 3% degradation in speed. It has been tested and approved by Compaq, Intel, NCR, Zenith, and the Department of Defense for use in high performance 80286 and 80386 Xenix or Unix based multi-user systems.

AT4 - 4 users\$795

AT8 - 8 users\$995

AT16 - 16 users\$1295

Phar Lap™ created the first tools that make it possible to develop 80386 applications which run under MS-DOS yet take advantage of the full power of the 80386. These include an 80386 monitor/loader that runs the 80386 in protected linear address mode, an assembler, linker and debugger. These tools are required for the MS-DOS version of the MicroWay NDP Compilers.

Phar Lap Tools\$495

PC/AT ACCELERATORS

287Turbo-10 10 MHz\$450

287Turbo-12 12 MHz\$550

287TurboPlus-12 12 MHz\$629

FASTCACHE-286 9 MHz\$299

FASTCACHE-286 12 MHz\$399

SUPERCACHE-286\$499

MATH COPROCESSORS

80387-20 20 MHz\$895

80387-16 16 MHz\$495

80287-10 10 MHz\$349

80287-8 8 MHz\$259

80287-6 6 MHz\$179

8087-2 8 MHz\$154

8087 5 MHz\$99

**Micro
Way**

The World Leader in PC Numerics

P.O. Box 79, Kingston, Mass. 02364 USA (617) 746-7341
32 High St., Kingston-Upon-Thames, U.K., 01-541-5466
St. Leonards, NSW, Australia 02-439-8400

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```

*----- include files -----*

; standard Mac definitions
Include MacTraps.D      ; condensed trap file
Include SysEqu.D        ; system equates
Include ToolEqu.D       ; toolbox equates
Include QuickEqu.D      ; toolbox equates

; our stuff
Include rectCDEFEqu.Txt ; private definitions for this file

*----- common entry point -----*

rectCDEFFProc

; provide a stack frame
LINK    A6, #-autoBytes ; set frame pointer, with enough
                        ; bytes for automatic variables

; save some registers
MOVEM.L D3-D7/A2-A4, -(SP) ; save some work registers

; the key to 68000 code : fill those registers
LEA      param(A6), A0      ; A0 points to param
MOVE.L   (A0)+, D3          ; D3 holds param ( usage varies )
MOVE.W   (A0)+, D7          ; D7 holds message (operation selector )
MOVEA.L  (A0)+, A2          ; A2 holds theControl (control record handle )
CLR.L    D4                 ; clear out D4
MOVE.W   (A0)+, D4          ; varCode into D4
MOVE.B   varFlagTable(D4), D4; D4 holds a byte of variation flags

; set a default function result of 0, while A0's pointing in the right direction
CLR.L    (A0)

; lock the control record down
MOVEA.L  A2, A0
_HLock

; get a pointer to the control record
MOVE.L   (A2), A2

; lock down and get a pointer to any control data block
MOVE.L   contrlData(A2), D0 ; grab the handle
BEQ      caseOut            ; jump if NIL handle

; we've got a control data block, so lock it
MOVEA.L  D0, A3             ; copy the handle
MOVEA.L  A3, A0             ; lock the block
_HLock
MOVEA.L  (A3), A3           ; get a pointer to the control data block

caseOut
; case out on the message
; just jump off a table of routine offsets
ADD.W    D7, D7             ; double the message integer
MOVE.W   messageTable(D7), D0; grab a table offset
JSR      messageTable(D0)    ; jump to messageTable + offset

; clean up and go home
; if there was a control data block, unlock it
MOVE.L   contrlData(A2), D0 ; grab the handle
BEQ      unlockRec          ; jump if NIL handle

; we've got a control data block, so unlock it
MOVEA.L  D0, A0             ; move the block's handle into place
_HUnlock ; and unlock it

unlockRec
; unlock the control record
MOVEA.L  theControl(A6), A0
_HUnlock

; restore the saved registers
MOVEM.L  (SP)+, D3-D7/A2-A4

; remove the stack frame
UNLK     A6

; fetch the return address
MOVEA.L  (SP)+, A0

; set stack pointer to the function result
ADD.L    #theResult-param, SP

; we're outta here
JMP      (A0)

*----- the message jump table -----*

; there are nine possible messages

```

(continued on next page)

Q. How many programmers does it take to maintain a MAKE dependency file?

A. NONE! If you use VersiMAKE™

VersiMAKE™ is a full-featured MAKE utility that includes:

■ Dependency Generation

Derives your system's dependencies, through analysis of its C and MASM source files. Say goodbye to manual maintenance of MAKE dependency files!

■ Wild Card File Name Matching

Analyzes an entire collection of source files with a single statement.

■ Nested Include File

Handles standard C and MASM "include" conventions, and the INCLUDE environment variable.

■ Powerful Macro Facilities

Built-in macros, user-defined macros, and environment variables.

■ Analytical Reports

Shows the entire Include file hierarchy for each source file analyzed, and all of the parent source files for each Include file.

Q. How many programmers does it take to trace a symbol thru your system?

A. ONE! If you use VersiCREF™

VersiCREF™ is a unique utility that creates a Master Cross-Reference of your entire system.

■ Multi-Lingual

Handles C, assembler, or both.

■ Flexible

File names with line numbers, or file names alone. Global and local symbols, or globals alone.

■ Powerful

Easily handles systems containing 100 source files or more.



VersiMAKE™ \$125
VersiCREF™ \$75
Both \$150

(Outside U.S., add \$5 for shipping and handling)

800-334-4096

(In NJ, 609-871-0202)

MC/VISA/AMEX accepted

SUMMIT INFORMATION SYSTEMS, INC.

73 East Lane, Willingboro, NJ 08046

CIRCLE NO. 146 ON READER SERVICE CARD

Only one language has the sleek design of C, the engineering features of Ada, and optimizing compilers that generate .OBJS highly compatible with Microsoft C:

MODULA-2

PMI is the principal supplier of tools for Modula-2, including:

★ **Repertoire®**: An enormous general-purpose toolkit. Includes hundreds of low-level routines and 5 high-level subsystems: (1) Industry's most powerful screen design/display system; supplements MS Windows; provides multi-line input fields, input validation, context sensitive help, forms, etc. (2) Sophisticated list-oriented DBMS; supports binary objects, variable-length records, garbage collection, damaged file recovery, etc. (3) Simple text editor; (4) Expression evaluators; (5) LISP-like list manager. Includes full source (over 600K), and 300p manual. ... **\$89**

★ **Repertoire®/Btrieve® Toolkit**: Novell/Soft-Craft's Btrieve file manager is the standard for large business applications. R/BT is a massive system for building Btrieve applications with Repertoire's screen system. Includes a customizable prototype application. Ideal for consultants. Includes full R/BT source and Repertoire object code. **\$149**

★ **EmsStorage™**: Advanced, high-level memory manager that detects and uses LIM Expanded Memory if present, or DOS memory if not. Provides automatic garbage collection and MS-Windows-like interface (lock/unlock functions) for porting programs to MS Windows. **\$49**

★ **Macro2**: Brings the full power of C's macro preprocessor to Modula-2; provides DEFINE, UNDEFINE, IFDEF, IFNDEF, INCLUDE, etc., for parameterized macro functions, conditional compilation, etc. Includes full source: **\$89**

★ **ModBase**: A full B+ tree DBMS that uses a file format compatible with Ashton-Tate's dBase III. Provides indexing and file manipulation routines for use independent of dBase; billions of records per file. Includes full source: **\$89**

★ **Graphix**: The Modula-2 interface to the widely used MetaWindow graphics library. Supports multiple fonts, mouse tracking, many printers (incl. Post-Script & LaserJet), over 30 display adapters, and hundreds of modes. Includes MetaWindow package. With source: **\$189**

Object only: **\$149**

Supported compilers: Logitech, SonyBrook, Fitted Software Tools, FTL, ITC, etc. All available exclusively from PMI; dealer inquiries welcome.

PMI

VISA/MC
AMEX/COD/PO

(503) 777-8844

BIX: pmi

4536 S.E. 50th
Portland, OR 97206

Telex: 6502691013

CIRCLE NO. 147 ON READER SERVICE CARD

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```
messageTable
DC.W doDrawCntl-messageTable ; draw the control
DC.W doTestCntl-messageTable ; test the control
DC.W doCalcCCntl-messageTable ; calculate the control's region
DC.W doInitCntl-messageTable ; do control initialization chores
DC.W doDispCntl-messageTable ; do control disposal chores
DC.W doPosCntl-messageTable ; reposition & update the control
DC.W doThumbCntl-messageTable ; calculate control dragging params
DC.W doDragCntl-messageTable ; drag the control
DC.W doAutoTrack-messageTable ; do the control's action proc

*----- the variations flag table -----*

; there are sixteen control variations

; eight bits are used to flag eight qualities of a control variation
; from bit 7 (hi) to bit 0 (lo), the bits are symbolically named :
; textBit - pictBit - iconBit - outBit - shadBit - bareBit - invBit - chngBit

varFlagTable
DC.B $10010010 ; text - outlined - invert
DC.B $10010001 ; text - outlined - content change
DC.B $10011010 ; text - shadowed - invert
DC.B $10011001 ; text - shadowed - content change

DC.B $01000110 ; pict - bare - invert
DC.B $01000101 ; pict - bare - content change
DC.B $01010010 ; pict - outlined - invert
DC.B $01010001 ; pict - outlined - content change
DC.B $01011010 ; pict - shadowed - invert
DC.B $01011001 ; pict - shadowed - content change

DC.B $00100110 ; icon - bare - invert
DC.B $00100101 ; icon - bare - content change
DC.B $00110010 ; icon - outlined - invert
DC.B $00110001 ; icon - outlined - content change
DC.B $00111010 ; icon - shadowed - invert
DC.B $00111001 ; icon - shadowed - content change

*----- doDrawCntl -----*

; the application wants the CDEF to draw the control
doDrawCntl
; if control is invisible, do nothing
TST.B ctrlVis(A2) ; non-zero if the control is visible
BEQ drawn ; it's invisible, so no need to draw it

; save the entry pen state
PEA entryPenState(A6)
_GetPenState

; normalize the pen state
_PenNormal

; save a copy of the entry clip region
CLR.L -(SP) ; get a new region
_NewRgn
MOVE.L (SP),entryClipRgnCopy(A6); copy the entry clip region into it
_GetClip

shadOutTest
; see if a shadowed outline is to be drawn
BTST #shadBit,D4 ; check it out
BEQ simpOutTest ; no shadowed outline, on to the simple outline test
BSR doShadOutline ; draw a shadowed outline
BSR shadOutIntClip ; set a clip rect for the interior
BRA interiorDrawing ; on to the interior tests

simpOutTest
; see if a simple outline is to be drawn
BTST #outBit,D4 ; check it out
BEQ noOutNoTest ; if not, on to the no-outline duties
BSR doSimpOutline ; draw a simple outline
BSR simpOutIntClip ; set a clip rect for the interior
BRA interiorDrawing ; on to the interior tests

noOutNoTest
BSR noOutIntClip ; set a clip rect for the interior

interiorDrawing
textTest
; see if the button will contain text
BTST #textBit,D4 ; text ?
BEQ pictTest ; no, so next test
BSR doTextInterior ; yes, so draw text interior
BRA drawn ; and jump on

pictTest
BTST #pictBit,D4 ; pict ?
```



```

BEQ  iconNoTest      ; no, so it's an icon button
BSR  doPictInterior  ; yes, so draw pict interior
BRA  drawn           ; and jump on

iconNoTest
BSR  doIconInterior  ; draw icon interior

drawn
; all drawn
; restore the entry pen state
PEA  entryPenState(A6)
_SetPenState

; restore the entry clipping region, and get rid of its holder
MOVE.L  entryClipRgnCopy(A6),-(SP)
MOVE.L  (SP),-(SP)
_SetClip
_DisposRgn

; so long
RTS

*----- doShadOutline -----*

; draw a shadowed outline for a button

doShadOutline
; move to the starting point for the horizontal shadow line: (left+2,bottom)
MOVE.W  contrlRect+left(A2),-(SP)
ADDQ.W  #0002,(SP)
MOVE.W  contrlRect+bottom(A2),-(SP)
_MoveTo

; draw a line to the right side of the horizontal shadow line: (right,bottom)
MOVE.W  contrlRect+right(A2),-(SP)
MOVE.W  contrlRect+bottom(A2),-(SP)
_LineTo

; draw a line to the top of the vertical shadow line: (right,top+2)
MOVE.W  contrlRect+right(A2),-(SP)
MOVE.W  contrlRect+top(A2),-(SP)
ADDQ.W  #0002,(SP)
_LineTo

; now draw an outline rect at (top,left,bottom,right)
PEA  contrlRect(A2)
_FrameRect

; all done
RTS

*----- doSimpOutline -----*

; draw a simple outline for a button

doSimpOutline
; draw an outline rect at (top,left,bottom,right)
PEA  contrlRect(A2)
_FrameRect

; all done
RTS

*----- shadOutIntClip -----*

; set the clip region for the interior of a shadowed outlined button

shadOutIntClip

; use the rect at (top+2,left+2,bottom-2,right-2)
MOVE.L  contrlRect+top(A2),interiorClipRect+top(A6)
MOVE.L  contrlRect+bottom(A2),interiorClipRect+bottom(A6)
ADDI.L  #00020002,interiorClipRect+top(A6)
SUBI.L  #00020002,interiorClipRect+bottom(A6)
PEA  interiorClipRect(A6)
_ClipRect

; all done
RTS

*----- simpOutIntClip -----*

; set the clip region for the interior of a simply outlined button

simpOutIntClip

; use the rect at (top+2,left+2,bottom-2,right-2)
MOVE.L  contrlRect+top(A2),interiorClipRect+top(A6)

```

(continued on next page)

PC/Forms Screen Management Software **SLASHES** Development Time!

- PC/Forms takes the hassle out of screen design, screen management and input data validation.
- Forms are created & maintained using a form editor, loaded and processed at run time via the PC/Forms run time library.
- This is not a code generator.
- There is no memory resident form manager.
- Forms can be from one to ten screens in length.
- Form dimensions are adjustable (for windowing).

Form Editor Features

- Full control over foreground & background video attributes.
- Access to the extended (graphics) char. set.
- Line and box drawing.
- Define and modify field attributes:
 - Field Name ▪ Field Order ▪ Edit Mask
 - Default ▪ Auto Tab ▪ Must Respond
 - Numeric Test ▪ Right Justify ▪ Echo Data
 - Display Only ▪ Upper Case ▪ Warning Only
 - Test Range ▪ Data Type ▪ Numeric Precision
- Test form utility.
- Generate program shell utility.
- Field reorder utility.
- Temporary exit to DOS.
- Compile form definitions to .OBJ files.

Run Time Library

- Routines are color (CGA, EGA, VGA) / monochrome independent.
- Forms are processed in dynamic memory.
- User written validation routines can be linked to fields.
- String, Byte, Integer, Long, Real, and Double data types are supported.
- Scrolling fields.
- Run time library source code included.
- Run time library includes (plus others):
 - load_form() ▪ put_field() ▪ get_form()
 - release_form() ▪ put_form() ▪ clear_form_buffer()
 - display_form() ▪ get_field() ▪ alter_field_attr()
- No royalties.

System Requirements

- IBM PC/XT/AT/PS2 or compatible.
- PC-DOS or MS-DOS 2.0 or later

Ordering Information

MC/VISA/Checks.
Demo disk available.
Call for shipping
dates on other ver-
sions.

Prices
Turbo Pascal . . . \$99.95
Microsoft Pascal . \$149.95
Microsoft C . . . \$149.95
Lattice C . . . \$149.95
Turbo C . . . \$149.95



**Golden
SOFTWARE**

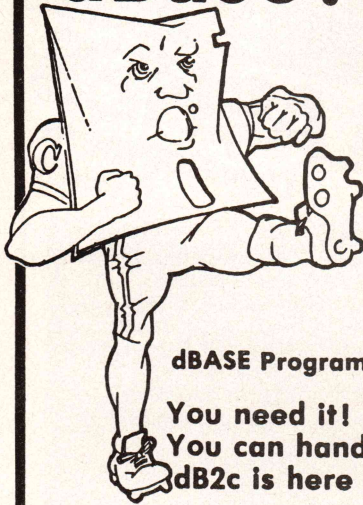
1-800-338-6754
(US)
1-216-292-0224
(OH)

P.O. Box 22216 • 23500 Mercantile Rd.
Beachwood, OH 44122

Hours: Mon-Fri: 7:30 a.m. - 4:30 p.m., EST.

CIRCLE NO. 148 ON READER SERVICE CARD

goodbye dBase!



dBASE Programmers

**You need it!
You can handle it!
dB2c is here now!**

dB2c Offers:

- Version 2.0 complete with Translator and File Handlers.
- Extensive implementation of dBASE III+ with over 200 functions and commands in C source code.
- Contains our own File Handlers plus interfaces for Lattice's dBC and Faircom's c-tree.
- Supports screen I/O with ANSI.SYS or fast assembler routines.
- Support for Microsoft, Lattice and Turbo C compilers.
- Tutor features of translation combined with familiar syntax of the library eases the transition to 'C'.
- One version supports MS-DOS, Xenix, Unix, OS-9 and Concurrent DOS.

**are you
ready?**

dB2c

Toolkit \$299



**DAVID J.
MARSH**

Call or Write:

**SOFTWARE
CONNECTION, INC.
POB 712, Ely, MN 55731
(218) 365-5097**

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```

MOVE.L    contrlRect+bottom(A2),interiorClipRect+bottom(A6)
ADDI.L    #$00020002,interiorClipRect+top(A6)
SUBI.L    #$00020002,interiorClipRect+bottom(A6)
PEA       interiorClipRect(A6)
_ClipRect

; all done
RTS

*----- noOutIntClip -----*

; set the clip region for the interior of a non-outlined button

noOutIntClip

; use the rect at (top,left,bottom,right)
MOVE.L    contrlRect+top(A2),interiorClipRect+top(A6)
MOVE.L    contrlRect+bottom(A2),interiorClipRect+bottom(A6)
PEA       interiorClipRect(A6)
_ClipRect

; done
RTS

*----- doTextInterior -----*

; draw a button's text content, in an indicated font

doTextInterior

; get a pointer to the current grafPort into A4
PEA       currentGrafPort(A6)
_GetPort
MOVEA.L    currentGrafPort(A6),A4

; we may be using a font other than the window's, so save current font settings
MOVE.L    txFont(A4),curFontAndFace(A6) ; font and style
MOVE.W    txSize(A4),curSize(A6) ; size

; the control's font is indicated by a value in the control record's contrlValue
; field: -1 for the window's font, 0 for the System font, 1 for the application
; font, anything else a standard Mac font number

; in the case of a font other than the System or window's font, the font style
; is in the ContrlMin field, and the font size is in the contrlMax field

; case out on the font
MOVE.W    contrlValue(A2),D0
BMI       useWindowsFont

; see if the new font is the System font or something else
TST.W     D0
BEQ       useSystemFont

useCustomFont
; set the font, style, and size for a font other than the System or window's font
MOVE.W    D0,txFont(A4) ; set the font
MOVE.W    contrlMin(A2),txFace(A4) ; set the style
MOVE.W    contrlMax(A2),txSize(A4) ; set the size
BRA       figgerFontInfo

useSystemFont
; set the font, style, and size for the system font (all zeroes will do it)
CLR.L     txFont(A4) ; set the font and style
CLR.W     txSize(A4) ; set the size

useWindowsFont
; we're using the window's current font, so font number, size, and style already set

figgerFontInfo
; so the font's set up -- let's get some more information
PEA       fontInfo(A6)
_GetFontInfo

; from that info, we can figure the vertical positioning for the button's text
; the equation: vertPos = rectBottom - (fontDescent + ((rectHeight-fontHeight)/2))
MOVE.W    interiorClipRect+bottom(A6),D0 ; bottom - top gives rectHeight
MOVE.L    D0,D7 ; bottom will be used again
SUB.W     interiorClipRect+top(A6),D0
SUB.W     fontInfo+ascent(A6),D0 ; then subtract fontHeight
SUB.W     fontInfo+descent(A6),D0
ASR.W     #1,D0 ; divide what's left by 2
ADD.W     fontInfo+descent(A6),D0 ; add it to the descent
SUB.W     D0,D7 ; then subtract it from bottom

; determine whether we'll be drawing the control's title or a STR resource
; if the control hilites via a content change and is hilited and there's a handle to
; the STR resource, we draw the STR resource

```

(continued on page 75)

PAINLESS WINDOWS.

Windows. Data Entry. Menus.
Finally, a C programmers' tool that makes
them as easy to use as *printf()*.
With Greenleaf DataWindowsTM,
you move in quantum leaps!

Snazzy Window Treatments

DataWindows represents an important breakthrough in C programming tools. It sets you free so you can create exciting programs quickly and easily, saving both time and money! Developed to work with the IBM PC, XT, AT, compatibles, and MSDOS or PCDOS, DataWindows is a carefully tooled system of C functions which will jazz up your programs with unprecedented efficiency.

Greenleaf DataWindows is integrated windows, transaction data entry, pop-up, pull-down, and Lotus style menu systems with:

- **Screen Management.** You don't have to remember what's on the display or the sequence in which you put it there. DataWindows does the grunt work. There are no restrictions.
- **Transaction Data Entry.** Data entry windows can have any number of fields with sophisticated options for reading many data types. Calls are made to help, validation, and other functions. Full featured text editing, protected and mandatory fields, dBASE type picture strings, context sensitive help, validation of fields and transactions, redefinable keys, password entry, attribute control, keyboard idle and much more.
- **Device Independence.** It detects the type of display adapter your computer is using and adjusts to it automatically for CGA, EGA, or monochrome. Logical video attributes are easy to use for color or monochrome.
- **Compatibility.** Runs with Microsoft Windows and IBM TopView.
- **The Greenleaf Tradition of Quality.** Reliable products. Professional documentation that gets you up and running quickly and keeps you there. Reference card. Newsletter and Bulletin board.

IBM, Microsoft & dBase, are registered trademarks of International Business Machines, Microsoft Corporation & Ashton-Tate respectively. PCDOS, IBM PC, XT, AT, & TopView are trademarks of IBM; MSDOS and Microsoft Windows are trademarks of Microsoft Corporation.



Stop Window Shopping

Order Today. Or call toll free for a free demo of the windows library that makes all the others obsolete.

Order any of these high performance tools by calling your dealer or 1-800-523-9830 today. Specify compiler when ordering. Add \$8 for UPS second day air, or \$5 for ground. Texas residents add sales tax. MasterCard, VISA, P.O., check, COD. In stock, shipped next day.

Greenleaf DataWindows	\$225
DataWindows Source Module	\$225
The Greenleaf Comm Library v2.0	\$185
The Greenleaf Functions v3.0	\$185
Digiboard Comm/4-II	\$325
Digiboard Comm/8-II	\$535



GREENLEAF
Software®

16479 Dallas Parkway, Suite 570
Dallas, TX 75243

Call Toll Free
1-800-523-9830

In Texas and Alaska, call
214-248-2561

Window Dressings

■ **Simple or Complex Windows.** Up to 254 powerful overlaid windows simultaneously, all with just one kind of window to remember! Yet any window can be from one character to 32K!

■ **Easy Window Operations.** DataWindows lets you move, zoom, frame, title, change colors, titles, frames, size, location, and make windows visible or invisible at will! Functions set cursor, attributes, and write data to any window or "current window". Word wrap, auto scroll, keyboard functions.

■ **Write to Any Window Any Time.** Windows may be visible, overlaid, or invisible, and you can write to them anyway. What you write will be seen when the windows become visible.

■ **DataWindows is fast!** It writes directly to video memory (in some modes).

■ **Easy to save!** Any window, complete with attributes, can be saved on disk quickly and efficiently.

■ **Source code available. No royalties.**

Also from Greenleaf:

The Greenleaf Functions v3.0

The most complete, mature C language function library for the IBM PC, XT, AT and close compatibles. Includes over 225 functions — DOS, disk, video, color text and graphics, string, time/date, keyboard, disk status and Ctrl-Break functions plus many more.

The Greenleaf Comm Library

Our 2.0 version is the hottest communications facility of its kind. Over 120 functions — ring buffered, interrupt driven asynchronous communications for up to 16 ports simultaneously with XMODEM, XON/XOFF, many many sophisticated features.

We support all popular C compilers for MSDOS/PCDOS: Microsoft, Lattice, Computer Innovations, Aztec, DeSmet, and others.

ALL GAIN, NO PAIN

Blow away the 640K barrier

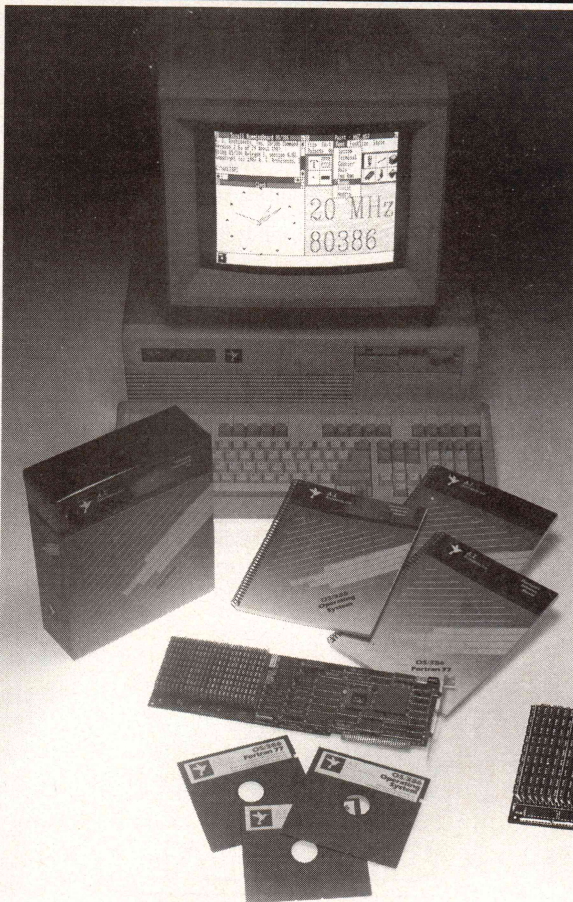
Gain the benefits of protected mode the easy way with OS/286™ and OS/386™. These tools for C, Fortran, Pascal and Assembly language programmers permit rapid conversion of existing DOS applications from "real" 8086 mode to "protected" 286 and 386 mode. They don't replace or modify DOS, but extend it to protected mode.

OS/286 and OS/386 are the only DOS extenders that span both the 286 and 386 processors, with 32-bit capability *today* on 386s that yields twice the performance of 16-bit mode. OS/286 and OS/386 have quickly become the preferred solution for developers of high performance, memory-intensive applications, including CADKEY, CASE, and Gold Hill, and premier language developers Lahey, and Metaware.

Our optional TOUCHDOWN™ BIOS supplement provides fast and reliable protected mode operation on *any* 286 system, even those with problems resetting the 286. (Ever notice how few existing machines Operating System/2 runs on?)

If your applications are running out of memory or need more speed, don't wait for the "solution" that means abandoning your investment in DOS. Enhance them now with OS/286 and OS/386 — *products not promises.*

MAKE BIG PROGRAMS RUN FASTER IN PROTECTED MODE



OS/286™ & OS/386™ Benefits:

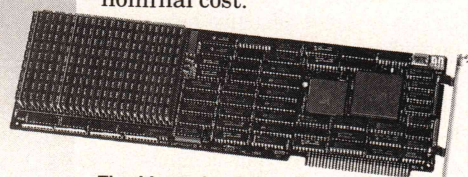
- Gain multi-megabytes of directly addressable memory (15-Mb-286, 4Gb-386)
- Increase performance by eliminating overlays and EMS
- Convert in days, not months
- Continue to work with a DOS interface and use existing TSRs, device drivers, graphic routines, etc.
- Stay compatible with the widest array of systems

A.I. Architects Software Developers Kit **\$495**

includes full support for:

- MetaWare High C (16 & 32 bit)
- Professional Pascal (16 & 32 bit)
- Lahey F77L FORTRAN
- Microsoft C & Fortran 4.0,
- MASM, MS-Link
- Phoenix PLINK86
- Halo & GSS Graphics
- Pharlap 386: ASM/LINK
more to come

Run time licenses for OS/286 and OS/386 are available at nominal cost.



The HummingBoard™ turns any XT or AT into the fastest 386 system available. The dual processor architecture boosts performance significantly over comparable single processor systems or accelerator boards. Available with 2 to 24Mb RAM, 16 or 20Mhz speed, and 387 floating point coprocessor.



A.I.
Architects, Inc.

One Kendall Square, Cambridge, MA 02139

TEL (617) 577-8052 FAX (617) 577-9774

OS/286, OS/386 and HummingBoard are trademarks of A.I. Architects, Inc., High C and Professional Pascal are trademarks of Metaware, Inc., F77L FORTRAN is a trademark of Lahey Computer Systems, Inc., Microsoft and MS-DOS are trademarks of Microsoft Corp.

CIRCLE NO. 170 ON READER SERVICE CARD

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```

; content change ?
BTST    #chgBit,D4
BEQ     useTitle           ; no content change, so use control's title
; control hilites via a content change
; is it hilited ?
MOVE.B  contrlHilite(A2),D0
BEQ     useTitle           ; 0=active, not hilited, so use control's title
ADDQ.B  #1,D0
BEQ     useTitle           ; 255=inactive, not hilited, so use control's title

; control hilites via a content change, and it's hilited
; do we have a handle to the content change string ?
MOVE.L  firstRsrcHndl(A3),D5
BEQ     useTitle           ; no, so use control's title

useSTR
; okay, we'll be using a content change STR resource, and we have a non-NIL handle
; lock the resource, put a pointer to it into D5, and set a flag
MOVEA.L D5,A0              ; handle into A0
HLock                   ; lock the STR resource
MOVEA.L D5,A0
MOVE.L  (A0),D5            ; set a pointer to it
MOVE.W  #1,usingCCRsrc(A6) ; set a flag
BRA     setStrWidth

useTitle
; using the control's title
; put a pointer to the string into D5, and clear a flag
LEA     contrlTitle(A2),A0
MOVE.L  A0,D5              ; set a pointer
CLR.W   usingCCRsrc(A6)    ; clear a flag

setStrWidth
; now, the horizontal positioning: start by getting the button's string's width
SUBQ.L  #2,SP              ; room for function result
MOVE.L  D5,-(SP)           ; the string
_StringWidth

; now, use that width to get the horizontal positioning
; the equation: horzPos = rectLeft + ( rectWidth-stringWidth ) / 2
MOVE.W  interiorClipRect+right(A6),D0 ; get rect width ( right - left )
SUB.W   interiorClipRect+left(A6),D0
SUB.W   (SP)+,D0           ; subtract string width
ASR.W   #1,D0              ; divide what's left by two
ADD.W   interiorClipRect+left(A6),D0 ; and add in the left side of rect

; now we have the horizontal and vertical starting position for string drawing
; let's move there ...
MOVE.W  D0,-(SP)           ; the horizontal starting position
MOVE.W  D7,-(SP)           ; the vertical starting position
_MoveTo

; paint a button's interior's background, according to the button's hilite state
; and whether we're drawing a content change

; test for content change imminent
TST.W   usingCCRsrc(A6)    ; remember, we just set or cleared this flag above
BNE     cCIminent1        ; flag set, content change imminent

; test the hilite state
bkgHSTest1
MOVE.B  contrlHilite(A2),D7
BEQ     hSActive1          ; 0 indicates an active button

bkgHSTest2
ADDQ.B  #1,D7              ; 255 indicates an inactive button
BNE     hSHilit1          ; 1-253 indicates a highlighted button

hSInactive1
; the button is inactive, so paint interior background white

cCIminent1
; a content change is imminent, so paint interior background white

hSActive1
; the button is active, so paint interior background white
MOVEA.L (A5),A0
PEA     white(A0)
_PenPat

hSHilit1
; the button is highlighted, so paint interior background black

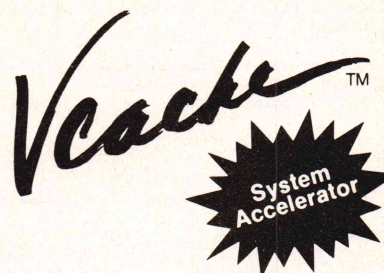
paintInteriorBkg
; paint the button's interior background
PEA     interiorClipRect(A6)
_PaintRect

; draw the button's interior's text, according to the button's hilite state
; and whether we're drawing a content change

```

(continued on next page)

Get Your Disks Moving!



Fast - increase your effective program speed up to 10 times! Data that has been read from your hard disk is retained in your computer's fast internal memory for super fast access the next time you need it.

Now Faster! - use the new Deferred Write option to cache disk Writes, for even faster processing.

Safe - use the Write Through option for maximum safety from system failures and power glitches.

Smart - Hot LRU algorithm identifies and prioritizes sector chains and single sector accesses; efficient hashing algorithm insures instant access to cached data.

Adaptable - use up to 15 megabytes of extended or expanded memory, or up to half a megabyte of standard memory. Cache multiple hard disks using standard BIOS interface or software driver.

Easy - installs with one statement; requires no modification of your existing programs; transparent to all normal programs.

Compatible - even with AutoCAD, removable media drives, disk partitioning software. . .

Includes - Vkette diskette accelerator, Vscreen mono accelerator, Vkey keyboard accelerator.

\$49.95 \$3 shipping/handling.
CA add 6% sales tax.

GOLDEN BOW SYSTEMS



2870 Fifth Avenue
Suite 201
San Diego, CA 92103
800/284-3269

Vcache operates with DOS systems. Vcache is a trademark of Golden Bow Systems. AutoCAD is a trademark of Autodesk Inc. DOS is a trademark of Microsoft Corp. and International Business Machines Corp.

CIRCLE NO. 151 ON READER SERVICE CARD

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```

; clear a flag that, if set, signals an inactive button
CLR.B    D6

; test for content change imminent
TST.W    usingCCRsrc(A6)
BNE      cCIImminent2      ; flag set, content change imminent

; test the hilite state
txtInHSTest1
MOVE.B    contrlHilite(A2),D7
BEQ      txHSActive2      ; 0 indicates an active button

txtInHSTest2
ADDQ.B    #1,D7            ; 255 indicates an inactive button
BNE      txHSHilit2      ; 1-253 indicates a highlighted button

txHSInactive2
; the button is inactive, so we'll draw the text in black,
;                                     then gray it out
ADDQ.B    #1,D6            ; set the inactive flag
BRA      drawText          ; and jump to draw

txHSHilit2
; the button is highlighted, so we'll draw the text in white
MOVE.W    #srcBic,txMode(A4) ; so the black bits show as white

cCIImminent2
; a content change is imminent, so we'll draw the text in black

txHSActive2
; the button is active, so we'll draw the text in black

drawText
; draw the button's string
MOVE.L    D5,-(SP)          ; pointer to the string to draw
_DrawString

; if we used a STR resource, unlock it
TST.W    usingCCRsrc(A6) ; are we using ?
BNE      grayTest          ; no, so jump ahead
MOVE.L    firstRsrcHndl(A3),A0 ; yes, so get the handle
_HUnlock          ; and unlock it

grayTest
; see if we've got an inactive button, in which case we gray the text
;                                     out
TST.B    D6
BEQ      txtGetNorm        ; not inactive, so no graying out

; yup, we're inactive, so let's get grayed out
; set the pen pattern to gray
MOVEA.L    (A5),A0
PEA      gray(A0)
_PenPat

; and set pattern transfer mode to ANDing with the inverse of the
;                                     pattern
MOVE #PatBic,-(SP)
_PenMode

; now paint the clip rectangle gray
PEA      interiorClipRect(A6)
_PaintRect

txtGetNorm
; get things back to normal, in case they were changed
; normalize the text transfer mode
MOVE.W    #srcOr,txMode(A4)

; restore the window's prior font settings
MOVE.L    curFontAndFace(A6),txFont(A4)
MOVE.W    curSize(A6),txSize(A4)

textDrawn
; text button interior's all drawn
RTS

*----- doPictInterior -----*

; draw a PICTure interior for the button

doPictInterior
; we'll draw in the picture's bounding rectangle
; we'll try to center this rectangle horizontally and vertically
; in the control's clipping rectangle
; if the control's too small, the icon lines up against
; the top and or left sides of the control's clipping rectangle

; move clipping rectangle's top and left coords into place
MOVE.L    interiorClipRect+top(A6),pictRect+top(A6)

; get the clipping rect's width into D6
MOVE.W    interiorClipRect+right(A6),D6
SUB.W     interiorClipRect+left(A6),D6

; get a pointer and handle to the picture
; if the control hilites via a content change and is hilited,
;                                     we draw the secondary
; PICT resource
; does the control hilite via a content change ?
BTST      #chngBit,D4
BEQ      usePictOne        ; doesn't hilite via a content
;                                     change

; is the control hilited ?
MOVE.B    contrlHilite(A2),D1
BEQ      usePictOne        ; 0=active, not hilited
ADDQ.B    #1,D1
BEQ      usePictOne        ; 255=inactive, not hilited

usePictTwo
; we'll be using the secondary PICT resource
; get a handle and set a flag
MOVE.L    secondRsrcHndl(A3),D5 ; the handle
BEQ      usePictOne        ; in case of a NIL handle
MOVE.W    #1,usingCCRsrc(A6) ; the flag
BRA      getPictPointer

usePictOne
; we'll be using the primary PICT resource
; get a handle and set a flag
MOVE.L    firstRsrcHndl(A3),D5 ; the handle
BEQ      pictDrawn        ; in case of a NIL handle
CLR.W     usingCCRsrc(A6) ; the flag

getPictPointer
; lock the PICT, and get a pointer to it
MOVEA.L    D5,A0
_HLock
MOVEA.L    D5,A4
MOVEA.L    (A4),A4          ; get a pointer

figPicWidth
; figure the picture's width
MOVE.W    picFrame+right(A4),D1
SUB.W     picFrame+left(A4),D1

; now subtract the picture's width from the clip width
SUB.W     D1,D6

; if 2 or more, divide by 2 and use as horizontal offset
; if it's < 2, no horizontal offset
CMPI.W    #2,D6
BLT.S     doPictRight

; divide and add the offset in
ASR.W     #1,D6             ; divide by two
ADD.W     D6,pictRect+left(A6)

; now do the right coord of rectangle
doPictRight
MOVE.W    pictRect+left(A6),pictRect+right(A6)
ADD.W     D1,pictRect+right(A6)

; get the clipping rect's height
MOVE.W    interiorClipRect+bottom(A6),D0 ; get clip height into D0
SUB.W     interiorClipRect+top(A6),D0

; figure the picture's height
MOVE.W    picFrame+bottom(A4),D1
SUB.W     picFrame+top(A4),D1

; now subtract the picture's height from the clip height
SUB.W     D1,D0

; if 2 or more, divide by 2 and use as vertical offset
; if it's < 2, no vertical offset
CMPI.W    #2,D0
BLT.S     doPictBottom

; divide and add the offset in
ASR.W     #1,D0             ; divide by two
ADD.W     D0,pictRect+top(A6)

; now do the bottom coord of rectangle
doPictBottom
MOVE.W    pictRect+top(A6),pictRect+bottom(A6)
ADD.W     D1,pictRect+bottom(A6)

; clear the background
PEA      interiorClipRect(A6)
_EraseRect

```



```

; okay, let's draw the picture (remember, it gets clipped to the button's interior )
MOVE.L    D5,-(SP)      ; the picture's handle
PEA       pictRect(A6)  ; the destination rectangle
_DrawPicture

; unlock the PICT
MOVEA.L D5,A0           ; handle's still here
_HUnlock

; now, adjust the picture for buttons that are hilited inverts or inactive
BSR       interiorAdjust

pictDrawn
; pict button interior's all drawn
RTS

*----- interiorAdjust-----*

; adjust a button's interior for buttons that are hilited inverts or inactive

interiorAdjust

; test the hilite state
hiliteTest1
MOVE.B    contrlHilite(A2),D0
BEQ       intAdjDone    ; 0 indicates a non-hilited active button

hiliteTest2
ADDQ.B    #1,D0         ; 255 indicates an inactive button
BNE       itsHilited    ; 1-253 indicates a highlighted button

itsInactive
; the button is inactive, so we'll gray it out
; set the pen pattern to gray
MOVEA.L   (A5),A0
PEA       gray(A0)
_PenPat

; and set pattern transfer mode to ANDing with the inverse of the pattern
MOVE #PatBic,-(SP)
_PenMode

; now paint the clip rectangle gray
PEA       interiorClipRect(A6)
_PaintRect

; leave
BRA       intAdjDone

itsHilited
; the button is highlighted
; test for being an invert
TST.W     usingCCRsrc(A6)
BNE       intAdjDone    ; d content changer, so we done

; the button's a hilited invert, so invert it
PEA       interiorClipRect(A6)
_InverRect

intAdjDone
; all done with the adjustment
RTS

*----- doIconInterior -----*

; draw an ICONic interior for the button

doIconInterior

; we'll draw in an icon-sized rectangle
; we'll try to center this rectangle horizontally and vertically
; in the control's clipping rectangle
; if the control's too small, the icon lines up against
; the top and or left sides of the control's clipping rectangle

; move clipping rectangle's top and left coords into place
MOVE.L    interiorClipRect+top(A6),iconRect+top(A6)

; get the clipping rect's width into D0
MOVE.W    interiorClipRect+right(A6),D0
SUB.W     interiorClipRect+left(A6),D0

; now subtract the icon width
SUBI.W    #iconSize,D0

; if 2 or more, divide by 2 and use as horizontal offset
; if it's < 2, no horizontal offset
CMPI.W    #2,D0
BLT.S     doIconRight

```

(continued on next page)

TRUE MULTITASKING

With

MultiDos Plus

"multitasking for the IBM-PC."

Ideal for developing applications in process control, data acquisition, communications, and other areas. Check these features which make **MultiDos Plus** an unbeatable value.

- Run up to 32 programs concurrently.
- Your software continues to run under DOS. No need to learn a new operating system.
- Use the compilers you already have. Supports software written in most languages.
- Operator commands to load/run programs, change priority, check program status, abort/suspend/resume programs.
- Programmatic interface via INT 15H for the following.
 - * Intertask message communication. Send/receive/check message present on 64 message queues.
 - * Task control by means of semaphores. Get/release/check semaphores.
 - * Change priority-256 priority levels.
 - * Suspend task for specified interval.
 - * Spawn and terminate external and internal tasks.
 - * Disable/enable multitasking.
 - * and more!
- Independent foreground/background displays.
- Access to DOS while applications are running.

Hardware/Software Requirements

IBM PC/XT/AT or true clone. Enough memory to hold **MultiDos Plus** (48 KB) and all your application programs. Also may need 4 or 16 KB memory for "hidden screens" for each active task. MS-DOS (or PC-DOS) 2.0 or later operating system.

only: **\$24.95** OR
\$99.95
with source code

Outside USA add \$5.00 shipping and handling.
Visa and Mastercard orders only call toll-free: 1-800-872-4566, ext. 350., or send check or money order to:

NANOSOFT
13 Westfield Rd, Natick, MA 01760
MA orders add 5% sales tax.

CIRCLE NO. 152 ON READER SERVICE CARD

function libraries
disassemblers
compilers
text editors
text filters
communications support
text formatters
interpreters
bulletin boards
co-routines
compiler compilers
window packages
assemblers
games
tutorials
math packages
link editors
languages
cross compilers
pre-processors
function libraries
disassemblers
compilers
text editors

The C Users' Group Library

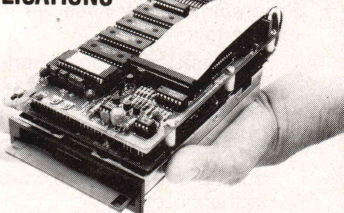
A Directory
of Public Domain
C Source Code

Send \$10
for Directory. Write
or call for more details
on over 100 volumes of
Public Domain C Source
Code.

The C Users' Group
P.O. Box 97
McPherson, KS 67460
(316) 241 1065

CIRCLE NO. 153 ON READER SERVICE CARD

COMPLETE DEVELOPMENT SYSTEM FOR MACHINE CONTROL APPLICATIONS



TINY188 is a low cost "PC somewhat compatible" engine for OEM controller applications. A selection of high level languages is available in ROM.

DDS188 An optional development board with EPROM programmer, floppy disk controller and added memory, removes to lower target system cost.

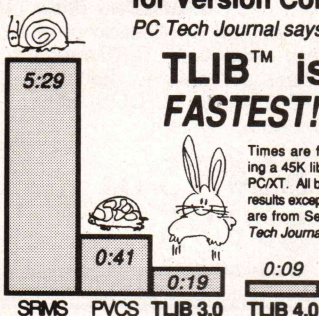
Prices start at \$269 each/\$99 at 1,000.

Vesta Technology, Inc.
(303) 422-8088

CIRCLE NO. 154 ON READER SERVICE CARD

for Version Control
PC Tech Journal says...

TLIB™ is FASTEST!



Times are for updating a 45K library on a PCXT. All benchmark results except TLIB 4.0 are from Sept 87 PC Tech Journal review.

"...packed with features... excellent..." Jim Vallino, *PC Tech Journal* Sept 87

"...has my highest recommendation." Ronny Richardson, *Computer Shopper* Aug 87

MS-DOS 2.x & 3.x Just \$99.95 + \$3 s/h Visa/MC

BURTON SYSTEMS SOFTWARE
P.O. Box 4156, Cary, NC 27519
(919) 469-3068

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```
; divide and add the offset in
ASR.W    #1,D0                      ; divide by two
ADD.W    D0,iconRect+left(A6)

; now do the right coord of rectangle
doIconRight
MOVE.W    iconRect+left(A6),iconRect+right(A6)
ADD.W    #iconSize,iconRect+right(A6)

; get the clipping rect's height
MOVE.W    interiorClipRect+bottom(A6),D0 ; get clip height into D0
SUB.W     interiorClipRect+top(A6),D0

; now subtract the icon height from the clip height
SUBI.W    #iconSize,D0

; if 2 or more, divide by 2 and use as vertical offset
; if it's < 2, no vertical offset
CMPI.W    #2,D0
BLT.S     doIconBottom

; divide and add the offset in
ASR.W    #1,D0                      ; divide by two
ADD.W    D0,iconRect+top(A6)

; now do the bottom coord of rectangle
doIconBottom
MOVE.W    iconRect+top(A6),iconRect+bottom(A6)
ADD.W    #iconSize,iconRect+bottom(A6)

; get a handle to the icon
; if the control hilites via a content change and is hilited, we draw the secondary
; ICON resource
; content change ?
BTST     #chgBit,D4
BEQ      useIconOne
; hilited ?
MOVE.B    contrlHilite(A2),D1
BEQ      useIconOne
ADDQ.B    #1,D1
BEQ      useIconOne

useIconTwo
; we'll be using the secondary ICON resource
; get a handle and set a flag
MOVE.L    secondRsrcHndl(A3),D5 ; the handle
BEQ      useIconOne             ; in case of a NIL handle
MOVE.W    #1,usingCCRsrc(A6)    ; the flag
BRA      clearBack              ; and jump ahead

useIconOne
; we'll be using the primary ICON resource
; get a handle and set a flag
MOVE.L    firstRsrcHndl(A3),D5 ; the handle
BEQ      iconDrawn              ; in case of a NIL handle
CLR.W     usingCCRsrc(A6)       ; the flag

clearBack
; clear the background
PEA      interiorClipRect(A6)
_EraseRect

; okay, let's draw the icon (remember, it gets clipped to the button's interior)
PEA      iconRect(A6)           ; the destination rectangle
MOVE.L    D5,-(SP)              ; the icon's handle
_PlotIcon

; now, adjust the icon for buttons that are hilited inverts or inactive
BSR      interiorAdjust

iconDrawn
; icon button interior's all drawn
RTS

*----- doTestCntl -----*

; the application wants CDEF to test a point to see if it's in an active control
doTestCntl

; first, see if the control's even active, or if it's in one of the two inactive modes
MOVE.B    contrlHilite(A2),D7 ;inactive control ?
CMPI.B    #inact254,D7
BEQ      setReturn2           ;yes, so return appropriate code
CMPI.B    #inact255,D7
BEQ      setReturn3           ;yes, so return appropriate code

; the control has been found to be active
; now find out if the supplied point is in the control's rectangle
SUBQ.L    #2,SP                ; room for the function result
```


TO THE MACS

```

MOVE.L    D3,-(SP)          ; param holds the mouse coords
PEA       contrlRect(A2)    ; pointer to the control's rectangle
_PtInRect

TST.B     (SP)+             ; scan result while chucking
BEQ       setReturn3        ; if point isn't in the control ...

; okay, the control's active, and the mouse point's in it
; the control has no separate parts, so we return the whole control's part number
setReturn1
MOVE.L    #wholePartNumber,theResult(A6)
RTS

; and here are the return codes for the other possibilities
setReturn2
MOVE.L    #254,theResult(A6) ;inactive type 254
RTS

setReturn3
MOVE.L    #0,theResult(A6)   ;inactive type 255
RTS                               ;or not in control

*----- doCalcCCntl -----*

; calculate the control's region

doCalcCCntl
MOVE.L    D3,-(SP)          ; param holds the waiting handle
PEA       contrlRect(A2)    ; use the control's rectangle
_RectRgn
RTS

*----- doInitCntl -----*

; do any special initialization of the control

; in this case, set up a control data block
; then load in any necessary resources, and store handles in the control data block

doInitCntl

; try to get a control data block
MOVE.L    #cntlDataBlokSize,D0
_NewHandle,CLEAR

; store the result
MOVE.L    A0,contrlData(A2)
BEQ       initDone          ; if we got no block, leave

lockDataBlock
MOVEA.L   A0,A3             ; save a copy of the block's handle
_Hlock    ; lock the block down
MOVEA.L   (A3),A3           ; get a pointer to the locked block

initTextTest
; is this is a text control ?
BTST      #textBit,D4        ; the test
BEQ       initPictTest      ; no, so jump on

; got a text control - does it indicate hiliting via a content change ?
BTST      #chngBit,D4        ; the test
BEQ       initPictTest      ; no, so jump on

; got a text control with content change hiliting, so load in the string resource and lock it
SUBQ.L    #4,SP             ; room for a handle
MOVE.L    #'STR ',-(SP)      ; the resource type
MOVE.W    contrlRFcon(A2),-(SP) ; the resource id
_GetResource ; try to grab that resource
MOVE.L    (SP)+,firstRsrcHndl(A3) ; store its handle (possibly NIL)
BRA       initDone

initPictTest
; see if this is a PICT control
BTST      #pictBit,D4        ; the test
BEQ       initIconTest      ; no, so jump on

; got a pict control, so load in the main PICT
SUBQ.L    #4,SP             ; room for a handle
MOVE.L    #'PICT',-(SP)      ; the resource type
MOVE.W    contrlRFcon+2(A2),-(SP) ; the resource id
_GetResource ; try to grab that resource
MOVE.L    (SP)+,firstRsrcHndl(A3) ; store its handle (possibly NIL)

isPictCC
; does it indicate hiliting via a content change ?
BTST      #chngBit,D4        ; the test
BEQ       initDone          ; no, so done

```

(continued on page 82)

MetaWINDOW

Power Graphics for your PC!

PC TECH JOURNAL

"Product of the Month"

"... a technological tour de force for fast PC graphics."

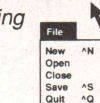
NO ROYALTIES!

MetaWINDOW is a
NEW! - QuickWINDOW/C
All the features of MetaWINDOW
for Microsoft Quick/C! - \$95*
and window managers.

Unparalleled Performance!

MetaWINDOW provides an expanded set of graphic drawing functions, plus the added functionality and performance required for designing multi-window desktop applications.

- auto-cursor tracking
- pull-down menus
- pop-up windows
- comprehensive graphic functions
- multiple fonts



10 Point 12 Point
Bold Italic

Enhanced Features!

- Display multiple bitmap or "filled-outline" fonts.
- Face fonts for bold, italic, underline or strike-out stylings.
- Full "RasterOp" transfer functions for writing, erasing, rubberbanding or dragging: lines, text, icons, bit images and complex objects.
- Create pop-up menus, windows and icons.
- Supports IBM's new PS/2 VGA and MCGA graphics.

MetaWINDOW comes complete with language bindings for 20 popular C, Pascal and Fortran compilers, plus dynamic runtime support for over 50 graphics adaptors and input devices.

MetaWINDOW

Advanced Graphics Toolkit
4 disks, 3 260 page manuals - \$195*

NEW! - TurboWINDOW/Pascal

All the features of MetaWINDOW for Borland Turbo Pascal Ver. 4! - \$95*
* Plus \$5.00 shipping and handling

TO ORDER CALL 1-800-332-1550
For information or in CA call 408-438-1550



METAGRAPHICS
SOFTWARE CORPORATION

269 Mount Hermon Road
Scotts Valley, CA 95066

CIRCLE NO. 155 ON READER SERVICE CARD

Turbo Tech Report Speaks Your Language.

Turbo Pascal
Articles and
Reviews

News and
commentary



A disk filled
with Turbo Pascal
code!

The newsletter/disk publication for Turbo Pascal® users

Are you looking for powerful utilities written in Turbo Pascal that you can use to develop software or incorporate into your programs? Are you interested in improving and expanding your Turbo Pascal programming skills?

Then you deserve a subscription to *Turbo Tech Report*, the bimonthly newsletter/disk publication from the publishers of *Dr. Dobbs's Journal* and *Micro/Systems Journal*. Each issue delivers more than 250K of Turbo Pascal source code programs on disk, and 24+ pages of articles, Turbo Pascal software and book reviews, and analysis and commentary.

It's the only publication delivering such focused technical articles with code on disk. Each valuable issue contains:

- **Articles** on topics like speedy 3D graphics, mathematical expression parsers, creating global gotos, memory resident and AI applications and more—all written by Turbo experts.

- **Reviews** of the latest Turbo Pascal software programs from companies like Borland International, Blaise

Computing, Media Cybernetics, Nostradamus, and more!

- **News and commentary** detailing the latest products and developments in the Turbo Pascal programming community.

- **A disk filled with Turbo Pascal code!** You'll get the Turbo Pascal utilities and routines discussed in the newsletter's articles, as well as applications developed by Turbo users from around the world. You'll receive programs that make labels, generate menus, provide faster screen access, transfer files, and more!

If you're an expert Turbo Pascal programmer or a novice interested in expanding your Turbo skills, you need a publication that speaks your language: *Turbo Tech Report*. Subscribe today at the special price of just \$99—that's 33% off the regular price of \$150. To order by credit card, call toll-free 1-800-533-4372. Or mail the coupon with your payment to *Turbo Tech Report*, 501 Galveston Drive, Redwood City, CA 94063.

—Yes! I want a one-year subscription to *Turbo Tech Report* (6 issues with 6 disks) for \$99.

Format: ☐ PC/MS-DOS ☐ Macintosh

CP/M: ☐ Kaypro ☐ Osborne ☐ Apple

PAYMENT MUST ACCOMPANY ALL ORDERS

☐ Check/money order enclosed.

☐ Charge my: ☐ VISA ☐ M/C ☐ AmExp.

Card # _____ Exp. _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

Orders outside U.S.: add \$30.

CIRCLE NO. 156 ON READER SERVICE CARD



Remember,

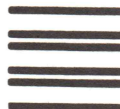
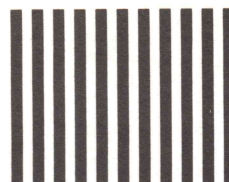


**Smart Buying
Decisions**



Start with DDJ

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

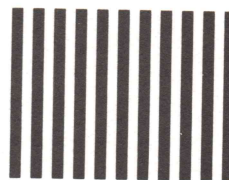
FIRST CLASS PERMIT #200, DALTON, MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

**Dr. Dobb's Journal of
Software Tools**
FOR THE PROFESSIONAL PROGRAMMER

Reader Service Dept.
P.O. Box 507
Dalton, Mass. 01227

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT #200, DALTON, MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

**Dr. Dobb's Journal of
Software Tools**
FOR THE PROFESSIONAL PROGRAMMER

Reader Service Dept.
P.O. Box 507
Dalton, Mass. 01227

Use this card for FREE, FAST information about the products and services listed in this issue. Simply circle the appropriate numbers below.

Name _____
 Title _____
 Company _____ Phone (_____) _____
 Address _____
 City/State/Zip _____

1	26	51	76	101	126	151	176	201	226	251	276	301	326	351	376
2	27	52	77	102	127	152	177	202	227	252	277	302	327	352	377
3	28	53	78	103	128	153	178	203	228	253	278	303	328	353	378
4	29	54	79	104	129	154	179	204	229	254	279	304	329	354	379
5	30	55	80	105	130	155	180	205	230	255	280	305	330	355	380
6	31	56	81	106	131	156	181	206	231	256	281	306	331	356	381
7	32	57	82	107	132	157	182	207	232	257	282	307	332	357	382
8	33	58	83	108	133	158	183	208	233	258	283	308	333	358	383
9	34	59	84	109	134	159	184	209	234	259	284	309	334	359	384
10	35	60	85	110	135	160	185	210	235	260	285	310	335	360	385
11	36	61	86	111	136	161	186	211	236	261	286	311	336	361	386
12	37	62	87	112	137	162	187	212	237	262	287	312	337	362	387
13	38	63	88	113	138	163	188	213	238	263	288	313	338	363	388
14	39	64	89	114	139	164	189	214	239	264	289	314	339	364	389
15	40	65	90	115	140	165	190	215	240	265	290	315	340	365	390
16	41	66	91	116	141	166	191	216	241	266	291	316	341	366	391
17	42	67	92	117	142	167	192	217	242	267	292	317	342	367	392
18	43	68	93	118	143	168	193	218	243	268	293	318	343	368	393
19	44	69	94	119	144	169	194	219	244	269	294	319	344	369	394
20	45	70	95	120	145	170	195	220	245	270	295	320	345	370	395
21	46	71	96	121	146	171	196	221	246	271	296	321	346	371	396
22	47	72	97	122	147	172	197	222	247	272	297	322	347	372	397
23	48	73	98	123	148	173	198	223	248	273	298	323	348	373	398
24	49	74	99	124	149	174	199	224	249	274	299	324	349	374	399
25	50	75	100	125	150	175	200	225	250	275	300	325	350	375	400

March '88: Use before June 30, 1988

1. Did you buy this issue on a newsstand? () Yes () No
2. Are you a subscriber? () Yes () No
3. Have you purchased a product as a result of seeing it advertised in *Dr. Dobb's Journal*? () Yes () No

**Dr. Dobb's Journal of
Software Tools**
FOR THE PROFESSIONAL PROGRAMMER

Use this card for FREE, FAST information about the products and services listed in this issue. Simply circle the appropriate numbers below.

Name _____
 Title _____
 Company _____ Phone (_____) _____
 Address _____
 City/State/Zip _____

1	26	51	76	101	126	151	176	201	226	251	276	301	326	351	376
2	27	52	77	102	127	152	177	202	227	252	277	302	327	352	377
3	28	53	78	103	128	153	178	203	228	253	278	303	328	353	378
4	29	54	79	104	129	154	179	204	229	254	279	304	329	354	379
5	30	55	80	105	130	155	180	205	230	255	280	305	330	355	380
6	31	56	81	106	131	156	181	206	231	256	281	306	331	356	381
7	32	57	82	107	132	157	182	207	232	257	282	307	332	357	382
8	33	58	83	108	133	158	183	208	233	258	283	308	333	358	383
9	34	59	84	109	134	159	184	209	234	259	284	309	334	359	384
10	35	60	85	110	135	160	185	210	235	260	285	310	335	360	385
11	36	61	86	111	136	161	186	211	236	261	286	311	336	361	386
12	37	62	87	112	137	162	187	212	237	262	287	312	337	362	387
13	38	63	88	113	138	163	188	213	238	263	288	313	338	363	388
14	39	64	89	114	139	164	189	214	239	264	289	314	339	364	389
15	40	65	90	115	140	165	190	215	240	265	290	315	340	365	390
16	41	66	91	116	141	166	191	216	241	266	291	316	341	366	391
17	42	67	92	117	142	167	192	217	242	267	292	317	342	367	392
18	43	68	93	118	143	168	193	218	243	268	293	318	343	368	393
19	44	69	94	119	144	169	194	219	244	269	294	319	344	369	394
20	45	70	95	120	145	170	195	220	245	270	295	320	345	370	395
21	46	71	96	121	146	171	196	221	246	271	296	321	346	371	396
22	47	72	97	122	147	172	197	222	247	272	297	322	347	372	397
23	48	73	98	123	148	173	198	223	248	273	298	323	348	373	398
24	49	74	99	124	149	174	199	224	249	274	299	324	349	374	399
25	50	75	100	125	150	175	200	225	250	275	300	325	350	375	400

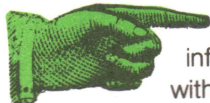
March '88: Use before June 30, 1988

1. Did you buy this issue on a newsstand? () Yes () No
2. Are you a subscriber? () Yes () No
3. Have you purchased a product as a result of seeing it advertised in *Dr. Dobb's Journal*? () Yes () No

**Dr. Dobb's Journal of
Software Tools**
FOR THE PROFESSIONAL PROGRAMMER

For Free Info ...

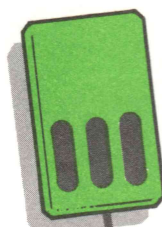
Start Here



Smart buyers start with *DDJ's* free information card, a shopping center filled with information about the products and services advertised in this very issue: everything from software and systems to peripherals and professional support services.

And smart buyers can use this free information card to quickly and easily gather a comprehensive file of facts, figures and product specs to sort out competing claims. Using *DDJ's* free information card can prevent you from making the wrong, costly buying decision.

Be a smart shopper. Complete and mail this postage paid card today!



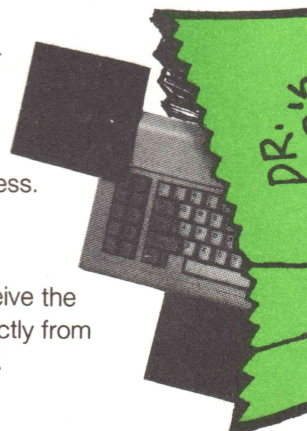
Take a Reader Service Card with You

It's Easy as ...

1. Circle the appropriate free information numbers, referring to the advertiser index for more information.

2. Fill in your name and address.

3. Mail today—postage is absolutely free. You'll receive the product information you need directly from the manufacturers, thanks to *DDJ*.



The Advertiser Index

Advertiser Name	Page #	RS#	Advertiser Name	Page #	RS#
AI Architects	74	170	Microsoft Press	115	184
Aker Corporation	103	*	MMC AD Systems	41	121
Alslys	85	160	Mortice Kern Systems, Inc.	105	176
American Cybernetics	99	172	MSJ Subscriptions	112	*
Andsor Research Inc.	62	137	Nanosoft Associates	77	152
ASI/American Software International	88	163	National Instruments	61	136
Aspen Scientific	39	*	Norton Utilities (The)	4-5	103
Austin Code Works	133	198	Norton Utilities (The)	38	119
Blaise Computing	2	102	Nu-Mega Technologies	67	144
Block Island Tech.	86	162	NWP Intellegent Solutions, Inc.	101	173
Borland International	1	101	Oakland Group, Inc. (The)	61	136
Borland International	9	105	Oakland Group, Inc. (The)	91	169
Breakpoint Computer Systems, Inc.	52	195	Oasys	46	122
Bryte Computer	60	135	Oregon Software	27	115
Burton Systems Software	78	*	Peacock Systems	65	142
C Store (The)	55	129	Peacock Systems	132	197
C Users Group	78	153	PMI	70	147
CAE/SAR Systems, Inc.	58	132	Polytron Corporation	15	108
Coders Source (The)	104	175	Pro Am Software	82	157
Coders Source (The)	111	181	Production Language Corp.	84	158
CNS, Inc.	26	114	Programmer's Connection	134-135	199
Cobalt Blue	65	141	Programmer's Paradise	107	178
Compaq Computer Corporation	44-45	*	Programmer's Shop (The)	87	165
Compu View	13	107	Programmer's Shop (The)	89	166-68
Computer Innovation	16-17	109	QCAD Systems, Inc.	64	140
Creative Programming	124	*	Quantum Software	25	112
Crosstalk Communications	C6	201	Quarterdeck Office Systems	21	110
DDJ Subscriptions	32	*	Raima Corporation	47	*
Desktop A.I.	88	164	Raima Corporation	plybg	*
Digital	29	116	Raima Corporation	106	177
Disk Software	86	161	SAS Institute	127	*
Ecosoft, Inc.	54	128	Scientific Endeavors	64	139
Elan Computer Group	60	134	Secom Information Products Co.	61	138
Elan Computer Group	66	143	Sharpe Systems Corporation	129	193
Essential Software	40	120	Sigs Publishing Inc.	53	127
Essential Software	48	123	SLR Systems	114	183
Fair-Com	110	180	Softfocus	128	192
Gimpel Software	83	*	Software Connection Inc.	72	149
Golden Bow Systems	75	151	Software Security, Inc.	57	131
Golden Software	71	148	Solution Systems	109	179
Greenleaf Software	73	150	Solution Systems	118	186
Interface Group, Inc.	92	*	Sophco	31	117
Lattice, Inc.	23	111	Springer-Verlag	102	174
Lugaru Software Ltd.	122	188	Summit Information Systems, Inc.	69	146
M&T Catalog of Books & Software Tools	plybg	*	Tenon Software	131	*
M&T Bound Volume 12 Promotion	plybg	*	Tool Makers (The)	126	191
Machine Independent Software Corp.	56	130	Turbo Power Software	122	189
Magna Charta Software	52	196	Turbo Tech Report	80	156
Manx Software Systems	7	104	VegaCon Corporation	26	113
Meridian Software Systems	49	124	Vermont Creative Software	59	133
Metagraphics Software Corporation	79	155	Vesta Technology Inc.	78	154
MetaWare Incorporated	120	187	Watcom C for IBM PCs	10-11	106
Micro Way	68	145	Watcom Waterloo C	51	125
Microsoft	93	171	Whitesmiths' Ltd.	C5	199
Microsoft	119-125	190	Whitewater Group (The)	35	118
Microsoft	117	185	Wyte Corporation	84	159
Microsoft Press	113	182			

*The advertiser prefers to be contacted by phone; consult ad.

Advertising Sales Offices

Midwest Charles Shively (415) 366-3600

Northeast Cynthia Zuck (718) 499-9333
Martha Brandt (415) 366-3600

Northern California/Northwest Lisa Boudreau (415) 366-3600

Southern California/AZ/NM/TX Michael Wiener (415) 366-3600

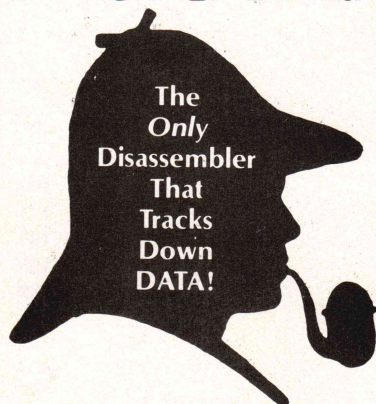
Telemarketing Rep./SE/SW USA Cheri Blum (714) 761-0294

Director of Marketing and Advertising Ferris Ferdon (415) 366-3600

NEW AND IMPROVED

Version 2

DISnDATa



- Fully disassembles both .EXE and .COM files!
- Flow- and Seg. Reg. Data-trace finds SEGs, PROCs, & Data Areas!
- Outputs SEGMENT & PROC pseudo-ops at proper places.
- Outputs data areas via proper form of DB/DW (ASCII text as strings, others as hex value).
- Labels both code & data. Labels of form 'Hxxxx' where 'xxxx' is hex offset from beginning of program.

NEW!

- User may easily input locations of multiple pgm. areas (if reqd.).
- 8086/88/186/286 op-codes, ('real' addressing mode).
- DOS function calls commented to show operation performed.
- Output format fully compatible with IBM*/Microsoft** assembler input.
- For IBM* PC*/XT*/AT* & compatibles, 128K+, DOS 2.0+.

#8634-22 PC-DISnDATa 2.0

(5¼" disk & manual) \$165

U.S. Funds only, drawn on a U.S. bank. Add \$3 shipping (U.S. & Canada), \$10 (overseas air) per item. Ohio residents please add local sales tax.

*Registered trademark, IBM Corporation.

**Registered trademark, Microsoft Corp.

To order, phone (513) 435-4480 (M-F, 9a.m.-5p.m., EST), or send check, money order, or VISA/MasterCard information (name, street address (No P.O. box please) card number, expiration date, and your telephone number) to:



PRO/AM SOFTWARE
220 Cardigan Road
Centerville, OH 45459
(513) 435-4480

Professional Software for
both Novice and Expert

TO THE MACS

Listing Three (Listing continued, text begins on page 90.)

```
; got a pict control with content change, so load in the secondary PICT
SUBQ.L #4,SP ; room for a handle
MOVE.L #'PICT',-(SP) ; the resource type
MOVE.W contrlRFcon(A2),-(SP) ; try to grab that resource
MOVE.L (SP)+,secondRsrcHndl(A3) ; store its handle (possibly NIL)
BRA initDone ; done

initIconTest
; no test needed; we have an icon control, so load in the main ICON
SUBQ.L #4,SP ; room for a handle
MOVE.L #'ICON',-(SP) ; the resource type
MOVE.W contrlRFcon+2(A2),-(SP) ; the resource id
_GetResource ; try to grab that resource
MOVE.L (SP)+,firstRsrcHndl(A3) ; store its handle (possibly NIL)

isIconCC
; does it indicate hiliting via a content change ?
BTST #chngBit,D4 ; the test
BEQ initDone ; no, so done

; got an icon control with content change, so load in the secondary ICON
SUBQ.L #4,SP ; room for a handle
MOVE.L #'ICON',-(SP) ; the resource type
MOVE.W contrlRFcon(A2),-(SP) ; the resource id
_GetResource ; try to grab that resource
MOVE.L (SP)+,secondRsrcHndl(A3) ; store its handle (possibly NIL)

initDone
; that's it for initialization
RTS

*----- doDispCntl -----0.*

; do any special disposal operations for the control

; in this case, release resources whose handles are stored in the control's
; data block, then release that block

doDispCntl
; see if we ever got a control data block
TST.L contrlData(A2)
BEQ dispDone ; no block, so leave

checkFirst
; see if there's a handle in the first slot
TST.L firstRsrcHndl(A3) ; got a real handle ?
BEQ checkSecond ; no, it's NIL, so jump ahead
MOVE.L firstRsrcHndl(A3),-(SP) ; handle okay, so let go of that resource
_ReleaseResource

checkSecond
; see if there's a handle in the second slot
TST.L secondRsrcHndl(A3) ; got a real handle ?
BEQ dropDataBlock ; no, it's NIL, so jump ahead
MOVE.L secondRsrcHndl(A3),-(SP) ; handle okay, so let go of that resource
_ReleaseResource

dropDataBlock
; now get rid of the control's data block
MOVEA.L contrlData(A2),A0
_DisposHandle

dispDone RTS

*----- doPosCntl -----*

; the position routine
; in this case, do nothing

doPosCntl RTS

*----- doThumbCntl -----*

; the thumb routine
; in this case, do nothing

doThumbCntl RTS

*----- doDragCntl -----*

; the drag routine
; in this case, do nothing

doDragCntl RTS

*----- doAutoTrack -----*
```



```
; the track routine
; in this case, do nothing
```

```
doAutoTrack
```

```
RTS
```

End Listing Three

Listing Four

```
*----- file information -----*
*
* rectCDEFEqu.Txt
*
* Private definitions for rectCDEF.Asm
*
* Edited with QUED/M 2.04
* Compiled under MDS 2.01
*
* Written and ©1987 by Stan Krute. All rights reserved. No part of this file,*
* or the object code it leads to, may be reproduced, in any form or by any *
* means, without the express written permission of the author and copyright *
* holder.
*
* Timestamp: 1:56 am EST September 29, 1987
* Spacstamp: 21E Halcyon Drive West Yarmouth, Massachusetts 02673
*
* This file looks good in 9 point Courier, QUED/M 2.04 tabs set to 3
*
*----- equates -----*

; stack frame offsets for function parameters
returnAddress EQU 4 ; return address' offset in frame
param EQU 8 ; for long-word-size parameter
message EQU 12 ; control message identifies desired operation
theControl EQU 14 ; calling control's handle's offset in frame
varCode EQU 18 ; which variation of the control
theResult EQU 20 ; function result offset in frame

; stack frame offsets for automatic (local) variables
entryPenState EQU -18 ; room to hold entry pen state (18 bytes)
currentGrafPort EQU -22 ; pointer to current grafPort ( 4 bytes )
curFontAndFace EQU -26 ; saved font number and style ( 4 bytes )
curSize EQU -28 ; saved font size ( 2 bytes )
fontInfo EQU -36 ; information about current font ( 8 bytes )
entryClipRgnCopy EQU -40 ; handle to copy of entry clip region (4 bytes)
interiorClipRect EQU -48 ; a clipping rectangle (8 bytes)
pictRect EQU -56 ; a PICTURE bounding rectangle (8 bytes)
iconRect EQU -56 ; an ICON bounding rectangle (8 bytes)
usingCCRsrc EQU -58 ; flags use of content change resource (2 bytes)
autoBytes EQU 58 ; size in bytes of automatic variable area

; hilite codes
inact254 EQU 254 ; hilite code to inactivate control
inact255 EQU 255 ; hilite code to inactivate control

; icon stuff
iconSize EQU 32 ; width and height of icon

; id's for our control definition
wholePartNumber EQU 10 ; part number for our whole control

; test bits
textBit EQU 7 ; if set, it's a text button
pictBit EQU 6 ; if set, it's a PICT button
iconBit EQU 5 ; if set, it's an ICON button
outBit EQU 4 ; if set, the button is outlined
shadBit EQU 3 ; if set, the button's outline is shadowed
bareBit EQU 2 ; if set, the button has no outline
invBit EQU 1 ; if set, the button shows hiliting via inversion
chngBit EQU 0 ; if set, the button shows hiliting via content change

; the control's data block
cntlDataBlokSize EQU 8 ; size of the control's data block
firstRsrcHndl EQU 0 ; offset of first data block field
secondRsrcHndl EQU 4 ; offset of second data block field
```

End Listing Four

Listing Five

```
; this file is called rectCDEF.Link

; ©1987 by Stan Krute -- all rights reserved

; timestamp: 12:57 am EST September 26, 1987
; spacstamp: 21E Halcyon Drive West Yarmouth, Mass. 02673

; turn off code listing to map file
]
```

(continued on next page)

Locate C Bugs before they Bite with PC-lint

PC-lint will analyze your C programs (one or many modules) and uncover glitches, bugs, quirks, and inconsistencies. It will catch subtle errors before they catch you. By examining multiple modules, PC-lint enjoys a perspective your compiler does not have.

"... a remarkable well thought-out product which will check for just about every conceivable coding error ... Its value increases with frequent use ... we confidently recommend PC-lint."

Andrew Binstock, *The C Gazette*

"PC-lint has everything going for it: flexibility, speed, good documentation, and a reasonable price. I exercised the product daily on a large, working, project to see if I could include it in my development tools. The answer is a definite YES."

Stephen D. Cooper, *Blue Notes San Francisco PC Users Group*

"... friendliness is a prime consideration in PC-lint. Gimpel has provided ways to make Lint shut up about all those errors you either know or don't care about. If Unix-Lint was implemented as well, I would use it more."

Don Malpass, *IEEE Software*

Gimpel Software

3207 Hogarth Lane
Collegeville PA 19426
(215)584-4261

PRICE: \$139.00 first copy, \$100 each additional, MC, VISA, COD, PA residents add 6% sales tax, Outside USA add \$15.

Runs on MS-DOS, works with any C compiler - direct support for 12 major C compilers including Microsoft 5.0, Turbo, C86+, Lattice, Datalight, Desmet
PC-lint is a trademark of Gimpel Software.

PRODUCTION QUALITY 68020 ANSI C Compiler

- Full 68020 instruction set
- Full 68881 support
- ANSI Standard reentrant library
- Extensive Sybolic Debug information
- Internal consistency checking
- Position independent code
- ROMable code

PC Hosted cross-compiler single user license	\$ 995.00
VME 68020 compiler single CPU license	\$ 1995.00
ANSI standard reentrant library source code	\$ 800.00

SPECIAL OFFER

For a limited time only we are offering full
library source FREE with the purchase of either
68020 C compiler

817-599-8366

P.O. Box 109, Weatherford, Texas 76086

CIRCLE NO. 158 ON READER SERVICE CARD

Conquer Time and Space.

Introducing XO-SHELL.™ Pop-Up Productivity for Programmers.

No matter what language you program in, XO-SHELL will help you hurdle the barriers to working faster and more efficiently by eliminating programming hassles. Only with RAM-resident XO-SHELL can you:

- DO CROSS-REFERENCING without leaving your editor
- VIEW ANY FILE and TRANSFER ANY SECTION into your editor or to your printer
- VIEW, COPY and ERASE files directly from a SCROLLABLE DIRECTORY DISPLAY
- With a single key stroke RETRIEVE previous DOS commands, then EDIT and REEXECUTE them
- DO SOURCE-LISTING while in your application
- OBTAIN KEY-CODES without a reference and without going through difficult interpretation
- INSERT GRAPHICS CHARACTERS in your source code.

XO-SHELL is for PCs, XTs, ATs, PS/2s, compatibles.



WYTE CORPORATION
701 Concord Avenue
Cambridge, MA 02138

\$49

plus \$5 shipping & handling
Call today toll-free
(800) 635-5011

In MA: (617) 868-7704
Visa, MasterCard

CIRCLE NO. 159 ON READER SERVICE CARD

TO THE MACS

Listing Five

(Listing continued, text begins on page 90.)

```
; the name of the input file is rectCDEF.Rel
rectCDEF.REL

; the name of the output file is rectCDEF
/Output rectCDEF

; set a type and creator for the output file
/Type '????' '????'

; end of Linker control file
$
```

End Listing Five

Listing Six

```
* this file is called rectCDEF.R0

* ©1987 by Stan Krute -- all rights reserved

* timestamp: 12:53 am EST September 26, 1987
* spacestamp: 21E Halcyon Drive West Yarmouth, Mass. 02673

* name of the output file is rectCDEF.Rsrc, type is RSRC,
creator is RSED

rectCDEF.Rsrc
RSRCSRD

* grab code resource 1 and turn it into a purgeable control
definition

TYPE CDEF = PROC
,40 (32)
rectCDEF
```

End Listing Six

Listing Seven

```
* this file is called rectCDEF.R1

* ©1987 by Stan Krute -- all rights reserved

* timestamp: 12:41 am EST September 26, 1987
* spacestamp: 21E Halcyon Drive West Yarmouth, Mass. 02673

* name of the input and output file is custom controls demo
!:custom controls demo

* grab CDEF resource from rectCDEF.Rsrc
INCLUDE :rectCDEF.Rsrc
```

End Listing Seven

Listing Eight

```
* this file is called rectCDEF.R2

* ©1987 by Stan Krute -- all rights reserved

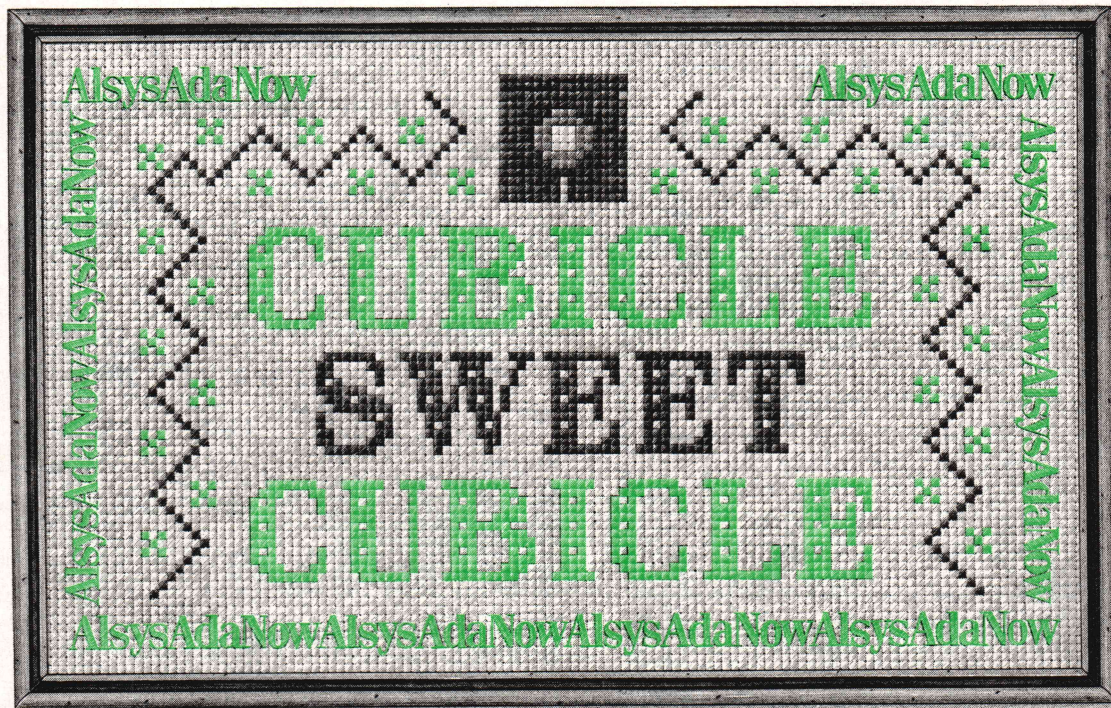
* timestamp: 12:41 am EST September 26, 1987
* spacestamp: 21E Halcyon Drive West Yarmouth, Mass. 02673

* name of the input and output file is custom controls demo PROJ
.rsrc !:custom controls demo PROJ.rsrc

* grab CDEF resource from rectCDEF.Rsrc
INCLUDE :rectCDEF.Rsrc
```

End Listings

Where do you find good Ada programmers?



Free
Poster :
see
coupon.

You could spend and spend trying to hire good Ada programmers and still not find what you need. Big demand; short supply. The irony is, your best Ada people may be the programmers you already have; all they need is good training.

Alsys offers a full range of quality Ada training products for growing your own programmers. For example...

■ A 27 video tape seminar covering the entire Ada language—18 hours of authoritative instruction by the principal designer of the language itself. Benefit? Understanding Ada's architecture and scope should be the foundation for all further work or study. It will help develop that most elusive skill: the Ada programming intuition to guess right.

■ For programmers ready for hands-on skills development, a comprehensive CAI course on a PC, running 50–60 hours, with exercises and progress tracking. Multiple

users. Licenses for 5 machines. The course is also excellent for brushing up, or extra work on one subject, or for new employees.

■ For practicing, and then moving directly to serious Ada programming, Alsys offers a full-featured, production quality Ada compiler, with tools, for the PC AT. This same compiler is used to build some of the largest Ada programs in existence!

Alsys offers more training products. A CAI course for programmers familiar with Fortran... a searchable, on-line version of the Reference Manual... a (limited) offering of live training courses.

Good Ada training for your own people. For Ada now. Write or call.

alsys

**Ada Programmers
Are Made — Not Hired.**

In the US: Alsys Inc., 1432 Main St., Waltham, MA 02154 Tel: (617) 890-0030

In the UK: Alsys Ltd., Partridge House, Newtown Rd., Henley-on-Thames, Oxon RG9 1EN Tel: 44 (491) 579090

In the rest of the world: Alsys SA, 29 Avenue de Versailles, 78170 La Celle St., Cloud, France Tel: 33 (1) 3918.12.44

Send me POSTER and more information on:

DDJ 3/88

____ *Ichbiah, Barnes & Firth on Ada 27-tape Video Series.*
____ *Lessons on Ada CAI Course.* ____ *Live Training.*
____ *PC AT Compiler and Tools.* ____ *AdaQuery On-Line Reference.*
____ *You Know Fortran, Ada is Simple CAI Course.*
____ *Ada Immersion Combination Package.*

Name _____

Company _____

Address _____

City _____ State _____ Zip _____

Phone () Ext. _____

Mail to: Alsys, Inc. 1432 Main St., Waltham, MA 02154

*Ada is a registered trademark of the U.S. Government (AJPO).

CIRCLE NO. 160 ON READER SERVICE CARD

Now in " C " !!!!!

Two and Three Dimensional Geometry

The added plus you need for developing sophisticated computer graphics, CAD, and programs that use computational geometry. You save time and money with its flexibility.

TurboGeometry Library

An excellent addition to Borland's Turbo Graphix Tool Box.

Over 150 ready to use two & three dimensional routines, such as Geometric Equations • Intersections • Curves • Arcs • Circles • Lines • 2&3 Dimensional Transforms • Polygons • Hidden Lines • Volumes • Perspectives • Clipping • Areas and many more..... Manual, full source code and sample programs. Only \$99.95US. Add \$5.00 for S&H in US. TexRes add 8% ST. 30 day guarantee. VISA, MC, MO, Chk PC(Comp), Turbo Pascal 2.0+, 4.0, & C. DOS 2.0+. When ordering, indicate language. ORDER YOUR LIBRARY TODAY!!

DISK SOFTWARE, INC., 2116 E. Arapaho, #487
Richardson, Texas 75081 (214) 423-7288

CIRCLE NO. 161 ON READER SERVICE CARD

Parallel Programming for "C"

INTERWORK™

A Concurrent Programming Toolkit

Interwork is a "C" program library which allows you to write your programs as a set of cooperating concurrent tasks. Very useful for simulation, real-time applications, and experimentation with parallel programming.

FEATURES

- Supports a very large number of tasks (typically more than 100) limited only by available memory. Low overhead per task results in very fast context switching.
- Provides a full set of inter-task communication (ITC) facilities, including shared memory, locks, semaphores, blocking queues, and UNIX™-style signals. Also has building blocks for constructing your own ITC facilities.
- Handles interrupts (DOS version) and integrates them into task scheduling. Supply your own interrupt handlers or block tasks on interrupts.
- Lets you trace task switches and inter-task communication.
- Comes with complete documentation including a user's manual and reference manual of commands.

Interwork is available for the following systems:

Hardware	Operating System	Price
IBM PC, XT, AT	PC-DOS 2.0 or later	\$129
IBM PC AT	XENIX™	\$159
DEC VAX™, SUN	UNIX 4.2BSD	\$249

PC-DOS version is compatible with DeSmet, Lattice, and Microsoft C compilers.

Please specify hardware and operating system when ordering. Shipping and handling included; COD orders add \$2.50. Send check or money order to:



Block Island Technologies
Innovative Computer Software

13563 NW Cornell Road, Suite 230, Portland, Oregon 97229-5892
(503) 241-8971

Trademarks: Interwork, Block Island Technologies; UNIX, AT&T Bell Laboratories, Inc.; XENIX, Microsoft, Inc.; VAX, Digital Equipment Corporation

CIRCLE NO. 162 ON READER SERVICE CARD

STRUCTURED PROGRAMMING

Listing One (Text begins on page 108.)

```

Program maddints;

{ This program illustrates addition of two 180 x 180 integer }
{ matrices A and B, storing the results in a similar matrix C }

Const max = 180;           { maximum columns/rows per square array }

Type arrayPtr = ^bigArray;
bigArray = array [1..max, 1..max] of integer;

Var A, B, C : arrayPtr;
    x, y : word;

{ ----- }

Procedure acquire (var D : arrayPtr);

{ Allocates the array on the heap and fills it with data }
{ This is a sample procedure for demo only. Replace it with }
{ the requisite code to acquire data from an external source }

Var r, c : word;

Begin
    New (D);                                     { allocate a node }
    For r := 1 to max do
        For c := 1 to max do
            D^ [r, c] := (r * 10) + c; { value of each array element }
End;
{ ----- }

Begin { main program }
    Acquire (A);                                { acquire array data }
    Acquire (B);
    New (C);                                    { set aside space for result }
    For y := 1 to max do
        For x := 1 to max do
            C^ [y, x] := A^ [y, x] + B^ [y, x]; { add arrays into C }
    Writeln ('Proof:');
    Writeln ('A [1, 1] = ', A^ [1, 1]);
    Writeln ('B [1, 1] = ', B^ [1, 1]);
    Writeln ('C [1, 1] = ', C^ [1, 1]);
    Writeln;
    Writeln ('A [max, max] = ', A^ [max, max]);
    Writeln ('B [max, max] = ', B^ [max, max]);
    Writeln ('C [max, max] = ', C^ [max, max]);
End.
    
```

End Listing One

Listing Two

```

Program hugemats;

{ Demo program to add two huge matrices > 64K, giving a third }

Const maxRows = 250;
      maxCols = 300;

Type dataObj = word;
      colPtr = ^colArray;
      colArray = array [1..maxCols] of dataObj;
      colNode = record
          col : colPtr;
      End;
      rowPtr = ^rowArray;
      rowArray = array [1..maxRows] of colNode;

Var A, B, C : rowPtr;
    x, y : word;
    error : Boolean;

{ ----- }

Procedure create (var D : rowPtr;
                  var error : Boolean);

{ Create huge array 'D' and pass back pointer to it }

Var row : word;

Begin
    Error := false;
    If maxAvail > sizeof (rowArray) then { if space available }
        GetMem (D, sizeof (rowArray)) { allocate row array }
    Else begin
        D := nil;
        Error := true;
    End;
    
```

(continued on page 88)

THE PROGRAMMER'S SHOP

helps save time, money, and cut frustrations. Compare, evaluate, and find products.

FREE Programmer's Update Magazine

"Keeping Professional Programmers Current with Developer's Technology." *Programmer's Update* lives up to its motto with a viewpoint unique to this industry. We use our experience and contacts to bring you articles about intriguing software trends and technical issues, interviews with authors and innovators, news about products, surveys, insightful commentary and predictions, valuable resource listings. You can get a FREE sample copy today by calling our toll-free number. Mention "DD388". A personal subscription is just \$25 a year.

Our Services:

- International Sales Desk
- Compare Products
- Help find a Publisher
- Evaluation Literature FREE
- Programmer's Update
- Dealers Inquire
- Newsletter
- Rush Order
- Over 700 products
- National Accounts Center

386 Development Tools

386 Assembler/Linker	PC	\$ 389
386 Debug - by Phar Lap	PC	\$ 129
386/DOS Extender	PC	\$ 919
DESQview PS/2	PC	\$ 109
F77L-EM - by Lahey	MS	Call
High C - by MetaWare	PC	Call
OS/286 & 386 by AI Architects	PC	Call
Paradox 386	MS	\$ 799

AI Languages

APT - Active Prolog Tutor - build applications interactively	PC	\$ 49
ARITY Prolog - full, 4 Meg		
Interpreter - debug, C, ASM	PC	\$ 229
COMPILER/Interpreter-EXE	PC	\$ 569
Cogent Prolog Compiler	MS	\$ 179
MicroProlog Prof. Comp./Interp.	PC	\$ 609
PC Scheme LISP - by TI	PC	\$ 85
Star Sapphire	MS	\$ 429
TransLISP - learn fast	MS	\$ 79
TransLISP PLUS	MS	\$ 149
TURBO PROLOG by Borland	PC	\$ 69
Others: IQ LISP (\$239), IQC LISP (\$269)		

Basic

BAS_C - economy	MS	\$ 179
BAS_PAS - economy	MS	\$ 135
Basic Development Tools	PC	\$ 89
db/Lib	MS	\$ 119
Exim Toolkit - full	PC	\$ 45
Finally - by Komputerwerks	PC	\$ 85
Inside Track	PC	\$ 49
Mach 2 by MicroHelp	PC	\$ 55
NetWorks by Exim	PC	\$ 89
QBase - screens	MS	\$ 79
QuickBASIC	PC	Call
Quick Pak-by Crescent Software	PC	\$ 59
Quick-Tools by BC Associates	PC	\$ 109
Stay-Res	PC	\$ 59
True Basic	PC	\$ 79
Turbo BASIC - by Borland	PC	\$ 69
Turbo BASIC Database Toolbox	MS	\$ 69

FEATURES

SoftTRAN, the Translation and Text Language by TransOptima - full procedural language like C plus pattern and nonprocedural constructs cuts development effort by up to 16 times. PC \$ 349

FORCE III, Dbase Compiler by Sophco-small .EXEs, user-defined functions, I/O directives through BIOS/DOS/ANSI/ FORCE/user-defined, extensions include FOR..NEXT loops, soundex, 1D arrays. Maverick. PC \$ 109

Program Objectively — for Less

Object-oriented programming tools free you from trivia tracking, make sophisticated applications practical quickly.

Consider **ACTOR**, full language and environment that lets you replace 100's of lines of code with just a few. It just may be the next standard language. Regularly \$419 (\$495 retail), **until March 31, 1988** only \$399.

CALL TODAY for free detailed literature about these and other object-oriented tools. Take advantage of these special prices — **order before March 31, 1988**. Mention "DD388."

RECENT DISCOVERY

CLARION DBMS by Barrington Systems.

Fast applications prototyping and development. Language, compiler, screen/report generators, editor, call other languages, read/write dBASE III+ files. PC, List: \$695

C Language-Compilers

AZTEC C86 - Commercial	PC	\$499
C86 PLUS - by CI	MS	\$359
Datelight Optimum - C	MS	\$ 99
Instant-C/16M	PC	Call
Lattice C - from Lattice	MS	\$259
Microsoft C 5.0- Codeview	MS	Call
Microsoft Quick C	MS	Call
Rex - C/86 standalone ROM	MS	\$695
Turbo C by Borland	PC	\$ 67

C Libraries-Files

BTree by Soft Focus	MS	\$ 69
CBTREE - Source, no royalties	MS	\$ 99
ctree by Faircom - no royalties	MS	\$315
rtree - report generation	PC	\$239
dB2C Toolkit V2.0	MS	\$249
dbQUERY - ad hoc, SQL-based	MS	Call
dbVISTA - Object only	MS	Call
Source - Single user	MS	Call
dBx - translator	MS	\$299

C-Screens, Windows, Graphics

C Worthy Interface Library	PC	\$249
Curses by Aspen Scientific	PC	\$109
dBASE Graphics for C	PC	\$ 69
ESSENTIAL GRAPHICS - fast	PC	\$185
FontWINDOW/PLUS	PC	\$229
GraphicC - new color version	PC	\$279
Greenleaf Data Windows	PC	\$155
w/source	PC	\$259
Terminal Mapping System	PC	\$279
TurboWINDOW/C - for Turbo C	PC	\$ 75
View Manager - by Blaise	PC	\$199
Vitamin C - screen I/O	PC	\$159
VC Screen	PC	\$ 79
Windows for C - fast	PC	Call
Windows for Data - validation	PC	Call
ZView - screen generator	MS	\$149

Atari ST & Amiga

We carry full lines of Manx & Lattice.

DBASE Language

Clipper compiler	PC	\$399
dBase III Plus	PC	\$429

Call for a catalog, literature, and solid value

800-421-8006

THE PROGRAMMER'S SHOP™

Your complete source for software, services and answers

5-D Pond Park Road, Hingham, MA 02043

Mass: 800-442-8070 or 617-740-2510 1/88

RECENT DISCOVERY

CO/SESSION - Remotely access PC and peripherals, train or trouble-shoot from off-site, 2 users on one program. Session record/playback, file transfer, terminal emulation, keyboard and voice modes. PC \$229

DBASE Language Cont.

dBASE III LANPack	PC	\$649
DBXL Interpreter by Word Tech	PC	\$ 99
FoxBASE+ Dev. - V2.0	MS	\$259
Quicksilver by Word Tech	PC	\$369

DBASE Support

dAnalyst	PC	\$219
dBase Tools for C	PC	\$ 65
dBrief with Brief	PC	Call
DBC III by Lattice	MS	\$169
dbug III	MS	\$179
Documentor - dFlow superset	MS	\$229
Genifer by Bytel-code generator	MS	\$279
QuickCode III Plus	MS	\$189
R&R Report Writer	MS	\$139
Seek-It - Query-by-example	PC	\$ 79
Silver Comm Library	MS	\$139
Tom Rettig's Library	PC	\$ 79
UI Programmer - user interfaces	PC	\$249

DataBase & File Management

CQL	PC	\$ 359
DataFlex by Data Access	PC	\$ 899
DataFlex multiuser	PC	\$1149
Magic PC	PC	\$ 699
Paradox - original	PC	\$ 369
Paradox V2.0	PC	\$ 469
Revelation by Cosmos	PC	\$ 779

Multilanguage Support

BTRIEVE ISAM	MS	\$185
BTRIEVE/N-multiuser	MS	\$455
GSS Graphics Dev't Toolkit	PC	\$375
HALO Development Package	MS	\$389
Graphics	PS	\$209
Help/Control - on line help	PC	\$ 99
HI-SCREEN XL	PC	\$129
HOOPS Graphics Library	PC	\$549
Informix 4GL-application builder	PC	Call
Informix SQL - ANSI standard	PC	Call
Instant Programmer's Help	MS	\$ 79
NET-TOOLS - NET-BIOS	PC	\$129
Opt Tech Sort - sort, merge	MS	\$ 99
Norton Guides	PC	\$ 75
Panel Plus	MS	\$395
Pfinish - by Phoenix	MS	\$209
Report Option - for Xtrieve	MS	\$109
Screen Sculptor	PC	\$ 89
SPSS/PC Plus	PC	\$749
Synergy - create user interfaces	MS	\$329
XQL - SQL for Btrieve	MS	\$649
Xtrieve - organize database	MS	\$179
ZAP Communications - VT 100	PC	\$ 89

Note: All prices subject to change without notice. Mention this ad. Some prices are specials. Ask about COD and POs. Formats: 3" laptop now available, plus 200 others. UPS surface shipping add \$3/item.

Make your C language
programs memory resident
with
DMS RESIDENT-C

Lattice

Microsoft

"hot-key"

enable

\$79.95

\$149.95
w/source



Make your assembler
programs memory resident
with
DMS RESIDENT-ASM

"hot key"

enable

\$79.95

\$149.95
w/source



American Software International
P.O. Box 523
Windsor, CT 06095-9998
(203) 688-5054

CIRCLE NO. 163 ON READER SERVICE CARD



Break The dBase Barrier

*dBase to C conversion
is now a reality*

Sooner or later you're going to run into the dBase wall. It may come up unexpectedly. Maybe you know it's there. But at some point you are going to need faster run-time, real portability, stronger code refinement, and source code security.

Using dBx to translate your dBase code to C is the perfect way to break the dBase barrier. C is portable, fast, and flexible. C programmers appreciate our commented, clean K&R code. If you are new to C, dBx is a great way to get up to speed. Why not call us today and discuss your individual situation.

dBx - The dBase to C Translator - It's Real

 **Desktop Ai** (203) 255-3400
303 Linwood Ave.
Fairfield, Ct., 06430
TELEX - 6502972226MCI

CIRCLE NO. 164 ON READER SERVICE CARD

STRUCTURED PROGRAMMING

Listing Two

(Listing continued, text begins on page 108.)

```

If D <> nil then
  For row := 1 to maxRows do begin      { allocate all rows }
    If not error then
      If maxAvail > sizeof (colArray) then      { if space }
        GetMem (D^ [row].col, sizeof (colArray)) { alloc row }
      Else
        Error := true;
    End;
  End;
End;
{ ----- }

Procedure acquire (var D      : rowPtr;
                  var error : Boolean);

  { Load data into array 'D' after creating it }

  Var   row, c : word;

Begin
  Create (D, error);
  If not error then
    For row := 1 to maxRows do
      For c := 1 to maxCols do
        D^ [row].col^ [c] := (row * 10) + c;  { modify to suit }
      End;
    End;
  { ----- }

Begin { main program }
  Writeln ('Size of each array is ',
          sizeof (rowArray) + (sizeof (colArray) * maxRows),
          ' bytes');
  Writeln ('Initial heap space = ', memAvail);
  Writeln ('Setting up array A');
  Acquire (A, error);

  If not error then begin
    Writeln ('Remaining heap space = ', memAvail : 6);
    Writeln ('Setting up array B');
    Acquire (B, error);
  End;

  If not error then begin
    Writeln ('Remaining heap space = ', memAvail : 6);
    Writeln ('Creating target array C');
    Create (C, error);
  End;

  If not error then
    Begin
      Writeln ('Remaining heap space = ', memAvail : 6);
      Writeln ('Adding arrays');
      For y := 1 to maxRows do
        For x := 1 to maxCols do
          C^ [y].col^ [x] := A^ [y].col^ [x] + B^ [y].col^ [x];
        End;
      Writeln;
      Writeln ('Proof:');
      Write ('A [1, 1] + B [1, 1] = C [1, 1] = ');
      Writeln (A^ [1].col^ [1] : 5, ' + ', B^ [1].col^ [1] : 5, ' = ',
              C^ [1].col^ [1] : 5);
      x := maxCols;
      y := maxRows;
      Write ('A [m, n] + B [m, n] = C [m, n] = ');
      Writeln (A^ [y].col^ [x] : 5, ' + ', B^ [y].col^ [x] : 5, ' = ',
              C^ [y].col^ [x] : 5);
    End
  Else
    Begin
      Writeln ('Out of memory: program ended');
      Write (#7);
    End;
  { beep }
End.

```

End Listings

THE PROGRAMMER'S SHOP

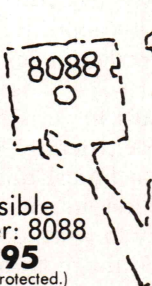
provides complete information, advice, guarantees and every product for Microcomputer Programming.

We've Added a New Teacher to Our Assembly Language Program.


Introducing The Visible Computer: 80286—an advanced version of the "Award Winning" TVC: 8088. Its animated simulation of the PC's microprocessor shows you exactly how assembly language instructions are carried out. And it comes complete with a comprehensive 400 p. manual.

TVC lets you choose from 45 demonstration programs that can be executed with the simulator—from simple register loads to advanced programs that manipulate interrupts and perform file I/O. It makes learning assembly language... elementary!





The Visible Computer: 8088
\$79.95
(Not Copy Protected.)
PS: \$69



The Visible Computer: 80286
\$99.95
(Not Copy Protected.)
PS: \$79

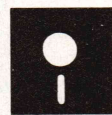
Software Masters™

The Visible Computer: 8088 for IBM PC/XT/AT and true compatibles; TVC: 80286 for IBM AT. To order direct: Software Masters, 2714 Finfeather, Bryan, TX 77801; (409) 822-9490. Include \$3.00 handling. Bank cards accepted.

CIRCLE NO. 166 ON READER SERVICE CARD

FIRMWARE DEVELOPMENT

C, MASM



LINK &
LOCATE



PROM



LINK & LOCATE enables PC users to produce ROM-based firmware for 8086/87/186 from object files generated by popular C compilers, such as from Microsoft, Lattice and Borland's Turbo C, and MASM from Microsoft. Provides full control of segment placement anywhere in memory. Supports output of Intel HEX file for PROM programmers, Intel OMF absolute object file for symbolic debuggers and in-circuit emulators. Includes Intel compatible linker, locator, librarian, hex formatter and cross reference generator. \$350.

NEW! Includes utility to support PC based PROM programmers.

Si SYSTEMS & SOFTWARE

PS: \$309

3303 Harbor Blvd., C11, Costa Mesa, CA 92626

Phone (714) 241-8650 FAX (714) 241-0377 TWX 910-695-0125

CIRCLE NO. 167 ON READER SERVICE CARD

*C Programmers: Combine C and COMMON LISP
to Increase the Power of Your Software*

with **TransLISP PLUS™** for Only \$99!

Simple.

Add LISP features to your software without making it a full time job. The TransLISP PLUS tutorial, on-line help, and 30 sample programs with commented source make it easy.

Practical.

Start by modifying the LISP sample programs and including them in a system you wrote in C. Yes, in C! TransLISP PLUS includes a C Language Interface that lets you integrate your Microsoft C code and libraries with all or portions of our LISP interpreter.

Use TransLISP PLUS to add natural or command language features to replace menus... or to flexibly manage related but disparate information. Code from C libraries provided by other vendors can be integrated into your program to perform tasks not normally part of LISP.

List \$318

Special Offer \$99

Offer good from 1/1/88 to 3/31/88

541 Main St., Suite 412, So. Weymouth, MA 02190

Thorough.

TransLISP PLUS took over 400 primitives from the most widely used and respected LISP standard, COMMON LISP, and made it available on IBM PCs, XT's, AT's, and virtually every other MSDOS machine. So now you can work with anything from a \$700 PC to a \$7000 PC.

The utilities toolbox is included at no charge with a built-in editor, pretty printer, cross reference, and additional debugging tools.

An optional Runtime encrypts your source code so that you can distribute your applications safely. You pay no royalties.

Requires MSDOS 2.0 +, 320K RAM, and a 360K floppy.

Royalty FREE Offer

This special offer from The Programmer's Shop includes the \$150 TransLISP Runtime. Create encrypted versions of your LISP source and distributed unlimited copies without paying royalties to The Coder's Source. You must order by 3/31/88 to take advantage of this special offer.

1-800-421-8006



**The
Coder's
Source™**

Call Today for FREE detailed information or try Risk-Free for 31 days, any product on this page.

HOURS: 8:30 A.M.-8:00 P.M. E.S.T.
5-DPond Park Road, Hingham, MA 02043
Mass: 800-442-8070 or 617-740-2510

800-421-8006

CIRCLE NO. 168 ON READER SERVICE CARD

THE PROGRAMMER'S SHOP

Your complete source for software, services and answers

MultiFinder, HyperCard, and More on Custom CDEFs

In case you missed January's introductory column: I'm here to talk Mac. Mostly I'll provide code samples. There'll also be discussion of programming tools, notable applications, conversations with programmers, and interesting ROM/OS topics.

This month I discuss three goodies from Apple (MultiFinder, HyperCard, and a manual) and finish up the custom control definition (CDEF) code project I began last time. If you hadn't noticed, the Macintosh universe is going supernova. Time to surf the flash. . .

MultiFinder Arrives

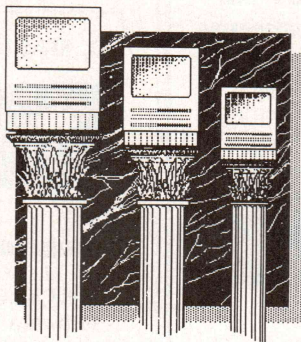
Official production copies of MultiFinder flew into my mountain aerie this month. (Although you will read this in February, I wrote it in October/November.) Bottom line: I love it. A clean bit of work. Thanks, Apple people. I run it almost all the time. The exception, when I'm print spooling to my ImageWriter. That'll change, probably by the time you read this.

Meantime, a few thoughts:

- Two Mbyte becomes the new minimum Mac RAM configuration. Think of the momentum Apple'd have if this chip-war silliness weren't happening. . .
- There may finally be a blessed stop to "What would you do with (fill in the blank) Mbyte of memory, anyway?" questions, now that the obvious answer becomes even more

by Stan Krute

- so: Fill it with programs and data that work intelligently with one another so as to further enlarge the universe of possible computer users.
- The major hole in this first version of MultiFinder: interprocess communication facilities. To raise the intelligence of the computing environ-



ment, applications must be able to talk to one another, both demo- and autocratically. Apple seems to understand this, so I figure this feature will show up in the next release. Plan ahead.

- Desk accessories, though supported, have lost a big chunk of their prime raison d'être: pseudomultitasking. We've now got semimultitasking, with the full thing lurking a few months ahead.

By the way, in case you don't know: with this release of MultiFinder, holding down the Option key when invoking a DA opens it in the foreground application's world rather than MultiFinder's separate desk accessory layer—useful for various parasitic DAs (spelling checkers, macro engines, et al.).

- The obvious hook into multitasking, GetNextEvent, has been used. If you guessed this three years ago, give yourself a no-prize.
- Following my favorite thread in the Apple mythos, MultiFinder favors decentralization of control. Rather than use a central multiprocessing dictator, MultiFinder gives applications the major say in CPU allocation. I like the implications.
- If you haven't already, say goodbye to low memory. Say goodbye to tricky OS hacks. Say goodbye to direct writing to the screen. Say goodbye to event and window manager fiddling. Assume as little as possible about anything. *Don't play no dirty tricks.*

Of course, capital-G Good Mac programmers never did funniness for fun, anyways. They did it to get

around some performance hole in the environment. So, OK, we won't do that anymore. But now it's up to Apple to make sure the necessary functionality gets built into the official toolkit. And while you're at it, Cupertinoans, how about a beefed-up Quickdraw/PostScript on a chip, please?

It's not too hard to get your programs "MultiFinder friendly." Call or write APDA for technical documentation, sample code, interfaces, and so on. I'll also try to work up something unique for the column. The docs I've got now are preliminary; by the time you read this, a lot more should be available.

HyperCard, Too

Another official arrival these past months: HyperCard. I like it a lot. We need a speedy and standard engine for accessing the gargantuan lased-out data piles that are looming, and HyperCard will do just fine.

HyperTalk, the HyperCard programming language, is simple, elegant, and powerful. Hits me as a nice synthesis of the history and state of computer languages, a sort of where-we-stand-today for the masses. I've done a fair amount of teaching programming, and this is a language I could use with all levels of learners.

There's a lot of the Atkinson snap to HyperCard, along with his attention to user interface and small implementation details. Quibbles? Well, there are the obvious things: You can only open a stack at a time. You can't resize the HyperCard window. There's a fair amount of modality and isolation from other applications. I have the feeling these things will get fixed.

Want to program the sucker? Play with it. Examine/modify scripts. Scan Apple's *HyperCard User Guide*. Then go on to Danny Goodman's fine work, *The Complete HyperCard Handbook* and APDA's *HyperCard Techni-*

Breakthrough in interface management. Generate C code from Dan Bricklin's Demo screens. Date fields. Full color support. Money fields. Fully programmable field behavior. Scrolling text within fields. Calculator style numeric input. User definable entry validation. Field marking. Orthogonal field movement. Specify fields by number or location. Source code included. Screen sizes limited only by memory. Interfaces with db_VISTA and other libraries. Text style numeric input. Input masking. List fields. Create spreadsheets. Includes Look & Feel screen designer. Integer fields. String formatting commands. Date and time validation functions. Generate C code with Look & Feel screen designer. Supports automatic vertical and horizontal scrolling. Clean screen fields per screen limited only by development. String fields. Easy to painting. Bind as much data as de data entry with commas. Ask a programming library. Hexadecimal or No fields. Float fields. Quick C. Speaker functions. Lattice. Create UNIX. Numeric validation. keystroke level. Customize screens 30 day money back guarantee. Gen assortment of editing commands. windows. Assign validation data to credentials. Pull down menus. Sup mode. All functions are kept in C style function reference. Pop-up functions. Numeric range checking. tive function names. Date and time Capture screens from existing as deep as desired. Easy to main checking. Date and time conver definition language based on C's ly definable borders. The current cally highlighted. Create reports. Convert old programs to C. Borders with titles. Color map enables use of logical colors. Toll-free telephone support line. 24 hour bulletin board. Automatically detects type of monitor being used. ANSI driver included. Screen and field definitions. Uses device drivers for portability. View windows. Read only fields. Rich assortment of editing commands. Pass- Includes ROM BIOS driver. Fields can support any data type. Scrolling/ included. Specify writeable and non-writeable positions within fields. borders. Se sor types. EGA, and UNIX. In for writing to about our ry. a variety functions. Multi-level Borders with eo RAM dri New device ated. Color ical use of with prompt grated help many screens data entry follow man customer sup higher level

C-scape 2.0

with

Look & Feel

The state-of-the-art interface management system preferred by professional C programmers and consultants worldwide.

Look & Feel

- WYSIWYG screen design tool
- Generates readable C code
- Create menus and data entry screens
- Define fields of any type
- Variables, prompts, and validation
- Line draw and erase
- Block, move, cut, paste, copy
- Horizontal and vertical scrolling
- Edit Dan Bricklin Demo slides
- Full color support
- Fast, easy, and fun to use
- Includes help
- Full-feature demo available

C-scape 2.0

- Windows, windows, windows
- Menus, menus, menus
- Vast help system
- Create any type of field
- Data entry and validation
- Smart borders
- Extensive function library
- Swappable device drivers
- Easy to learn and use
- Easy to maintain and modify
- Unsurpassed flexibility
- Professional manual
- No royalties; no run-time license
- Source code included
- Demo package available

OAKLAND

675 Massachusetts Avenue, Cambridge, MA 02139-3309

800-233-3733

CALL

617-491-7311

NOW



PC/MS-DOS \$299 (price includes C-scape, Look & Feel, source and manual). UNIX/others call. 30-day review.

readable C code. Portable. Easily modifiable functions. No royalties. Source code included. Apollo and Data General. Professional support. Interface examples for data base management. Validation at keystroke level. Vast integrated and indexed context-sensitive help system. Save and restore regions of the display. Now supporting Quick C, Turbo C, Lattice, Microsoft, UNIX, XENIX, and others. And that's not all. Call for demo.

The West Coast Computer Faire announces the first Computer Matchmaking Service.

You won't have to depend on fate at the 13th West Coast Computer Faire to find the products and services that are the perfect match for your needs.

We start you out on your path to high-tech bliss with Vertical Market Matching. We bring in the companies selling quality computers, software, peripherals, and add-ons—companies that meet the needs of people involved in specific business segments such as finance, medicine, manufacturing, law, education, engineering, and construction.

And our Product Matching makes it easy for you to find the software, add-ons and upgrades for the Commodore Amiga, Apple II or Macintosh, IBM PC/MS-DOS, IBM PS/2, Atari, Lotus and more, that will keep you happily gazing into your current system's eyes. Plus, we counsel you on the latest techniques and insights in our outstanding Conference sessions.

The West Coast Computer Faire has made and will make more matches than any another computer show. It's time we made the perfect match for you.

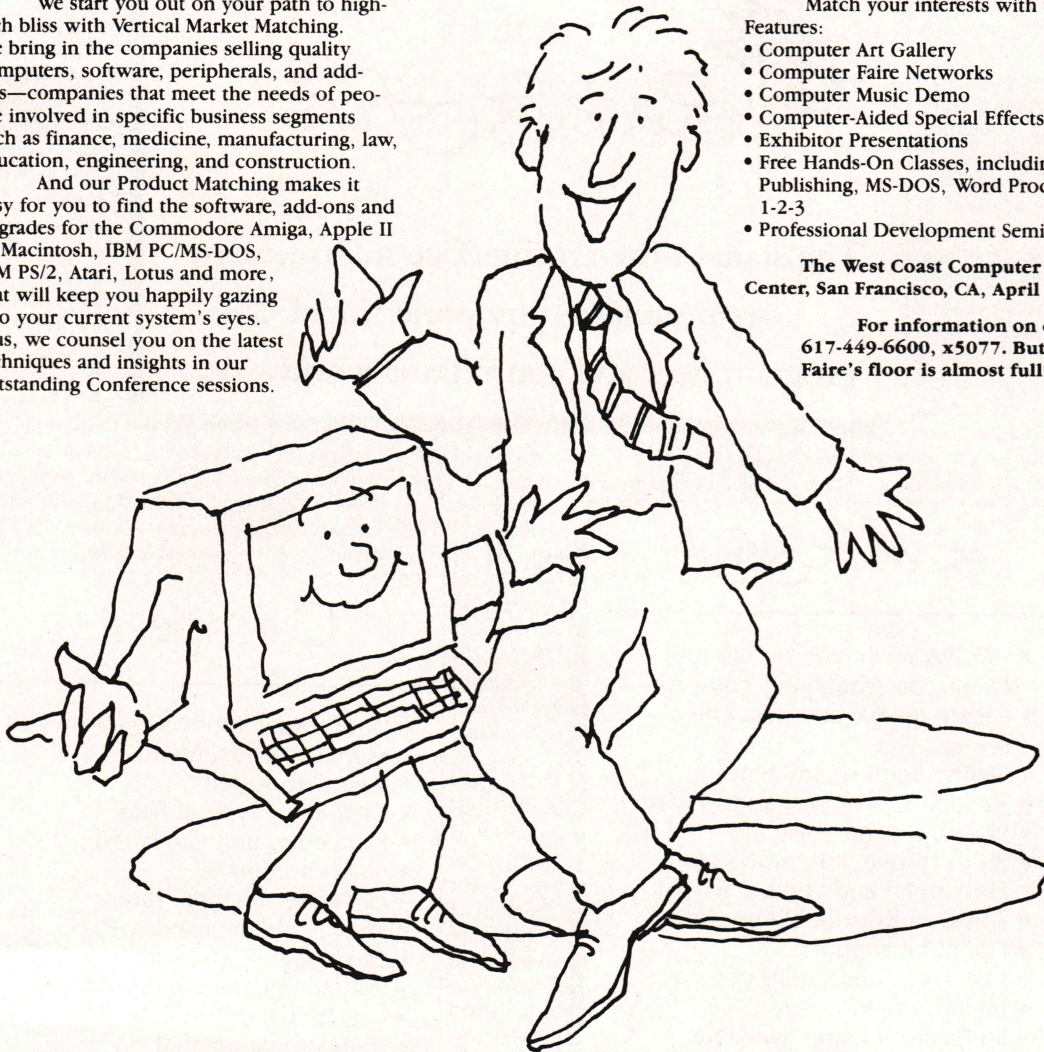
Match your interests with these Faire

Features:

- Computer Art Gallery
- Computer Faire Networks
- Computer Music Demo
- Computer-Aided Special Effects Demo
- Exhibitor Presentations
- Free Hands-On Classes, including Desktop Publishing, MS-DOS, Word Processing, Lotus 1-2-3
- Professional Development Seminars

The West Coast Computer Faire, Moscone Center, San Francisco, CA, April 7-10, 1988

For information on exhibiting, call 617-449-6600, x5077. But hurry — the Faire's floor is almost full!



Register early and save \$5!

Fill out this coupon and mail with your check(s), for \$15.00 for each registrant, postmarked by March 17th, 1988. Include the names and addresses of registrants for whom you are enclosing a check. (Photocopy coupon for additional registrants.)

Name _____ Title _____
Company _____
Address _____
City _____ State _____ Zip _____
Phone (____) _____

Four day conference and exhibits \$15.00 in advance. \$20.00 at the door. Make check payable to "The West Coast Computer Faire." Mail to: Attendee Registration Department, The West Coast Computer Faire, 300 First Avenue, Needham, MA 02194. Advanced registrations accepted only with full payment and each registrant's name and address. Tickets will be mailed to each individual registrant separately.

**THE 13th WEST COAST
COMPUTER FAIRE**

April 7-10, 1988, Moscone Center, San Francisco, CA

cal Reference Package. Also, Peter Olson, proprietor of Delphi's ICON-tact Mac forum, has written a sweet piece of shareware, Stackware Detective, to let you peek a little deeper. Check it out, and remember to pay if you end up using it.

My own HyperCard explorations are traveling along the XCMD and XFCN paths. These are hooks that let you add your own commands to HyperTalk. You can find details in the APDA docs. I'll be bringing you some samples in the near future. As PEEK and POKE are to BASIC, and slots are to the Apple II, the X twins are to HyperCard.

One of the Great Docs

Apple's always impressed me with its documentation, especially the programmers' docs. I think of the early Applesoft manuals, the Apple Pascal books, *Inside Macintosh*, Bo Three Bob's tech notes, and on and on. There have always been individuals at the company who care about this sort of thing, realizing its importance, and they've been able to marshal resources and produce.

There's an especially nice document out that you may have missed. It's one that all Mac programmers should read, ponder, and pay attention to. Its title is *Human Interface Guidelines: The Apple Desktop Interface*, and it's available from APDA.

I know, the title's not particularly exciting. But the content is, to me at least. Some of the more frustrating discussions I've had are with programmers who just don't understand why they should follow Apple's interface guidelines as closely as makes sense for their application. It's not because all Apple's decisions are cosmically correct. They're not. But they're darn close. And above all quibbling, consistent operational similarities between applications empower users. Period.

Please read this book, folks.

Code Corner: Custom CDEFs, Part 2

Resource Revelations

Minor problem: As noted last month, I use ResEdit to build most

of my Macintosh program resources. It's so Mac-like. But there's no text file script left over. So how can I reveal custom controls demo's resources to you?

Macintosh Programmer's Workshop (MPW) includes a Mac resource decompiler, DeRez. It takes a file's resources and produces a text file script description. It'd do the trick. But I don't use MPW (yet?). Cruising Delphi, my favorite on-line hangout, I found a fine solution: Alan Dahlbom's ResTools 2.01. It does much of what MPW's Rez and DeRez do, uses a compatible C-like syntax, and is free. A big thanks to

Alan for a fine piece of programming and a public service.

So, Listing One, starting on page 64, shows the ResTools 2.01 decompilation of most of the resources included in custom controls demo PROJ.rsrc. The syntax, as mentioned earlier, is C-like; if it doesn't make immediate sense, go to *Inside Mac* and stare at the particular resource's description.

I've removed the decompilation of the file's PICTs, ICONs, and ICN#s; these graphic resources look kind of silly as streams of hexits. Table 1, below, indicates where you can find images of the PICTs as well as the

Microsoft® University offers the only systems training straight from the source.

Microsoft University courses take you to the heart of our microcomputer software architecture. Our systems software curriculum combines in-depth technical presentations, problem-solving sessions, and practical hands-on workshops.

Microsoft University Winter Quarter

A	MS® OS/2 Programming Environment (4 days)	\$1000
B	MS OS/2 Advanced Programming (5 days)	\$1250
C	MS OS/2 Device Drivers (4 days)	\$1000 *
D	MS OS/2 LAN Manager API Programming (5 days)	\$1000
E	Microsoft Windows API Programming (5 days)	\$1000
F	Advanced Microsoft Windows API Programming (5 days)	\$1000
G	Programming in Microsoft C (5 days)	\$ 800
H	Microsoft Macro Assembler (MASM) Programming (4 days)	\$ 800

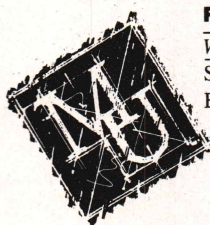
Courses held in both East and West Coast cities.

FEBRUARY 1988 SCHEDULE

Week of:	Feb. 1	Feb. 8	Feb. 15	Feb. 22	Feb. 29
SEATTLE	E,G,H	A,C,E	B,D,F,G	A,C,E	B,D,F,G
BOSTON**		G	E	B	F

MARCH 1988 SCHEDULE

Week of:	Mar. 7	Mar. 14	Mar. 21
SEATTLE	A,B,C,E	D,F,G	A,B,E,H
BOSTON**	D	C	



Tuition is per person and includes all course materials.

* For a limited time, this course also includes the Microsoft Device Driver Development Kit, a \$300 value, at no additional charge.

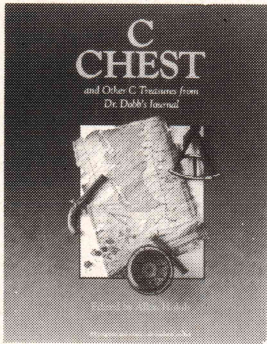
**Fees slightly higher in Boston.

Call the Microsoft University Registrar and use your credit card to enroll now.

(206) 882-8080.

Microsoft®

C Tools From M&T Books



C Chest and Other C Treasures from Dr. Dobb's Journal

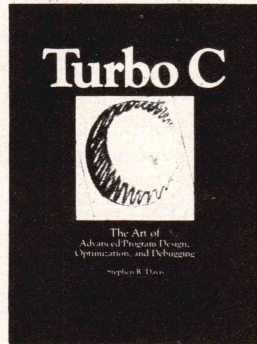
by Allen Holub

This comprehensive anthology contains the popular "C Chest" columns from *Dr. Dobb's Journal of Software Tools*, along with the lively philosophical and practical discussions they inspired, in addition to other information-packed articles by C experts.

Topics covered include: pipes, wild-card expansion, and quoted arguments; sorting routines; command-line processing; queues and bit maps; utilities such as *ls*, *make*, and *more*; expression parsing; hyphenation; IBM cursor control and an Fget that edits; redirection; accessing IBM video display memory; trees; an AVL tree database package; directory traversal; sets; shrinking .EXE file images; hashing, expressions, and Roman numerals; and statistical applications of digital low-pass filters.

All subroutines and programs are written in C and are available on disk with full C source code. MS-DOS format.

Book & Disk (MS-DOS)	Item #49-6	\$39.95
Book	Item #40-2	\$24.95



Turbo C: The Art Program Design, of Advanced Optimization and Debugging

by Stephen R. Davis

Overflowing with example programs this book fully describes the techniques necessary to skillfully program, optimize and debug in Turbo C. Every topic and Turbo C feature discussed is fully demonstrated in Turbo C source code examples. Advanced topics such as pointers; direct screen I/O; inline statements in Turbo C; and how to intercept and redirect BIOS calls are all covered indepth. The author further demonstrates these advanced topics by writing a RAM resident pop-up program in Turbo C. Learn about the differences between Unix C and Turbo C, about the transition from Turbo Pascal to Turbo C, and about the superset of K&R C features implemented in Turbo C and included in the proposed ANSI C standard.

Book & Disk (MS-DOS)	Item #45-3	\$39.95
Book	Item #38-0	\$24.95

Special Packages

CP/M C Package SAVE \$27!

Receive: Dr. Dobb's Toolbook of C, The Small-C Handbook, the Small-C Compiler, Small-Mac Assembler, and Small-Tools Text Processing Programs. Only \$99.95!

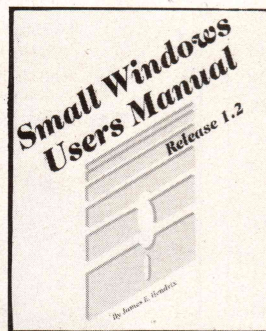
CP/M C Package	Item #005A	\$99.95
-----------------------	-------------------	----------------

MS-DOS C Package SAVE \$22!

Receive: Dr. Dobb's Toolbook of C, The Small-C Handbook and MS-DOS Addendum, the Small-C Compiler, Small-Windows, and Small-Tools Text Processing Programs. Only \$109.95!

MS-DOS C Package	Item #005W	\$109.95
-------------------------	-------------------	-----------------

C Disk Formats: Please specify MS-DOS or CP/M. For CP/M, specify: Apple, Kaypro, Osborne, Zenith Z-100 DS/DD, 8" SS/SD



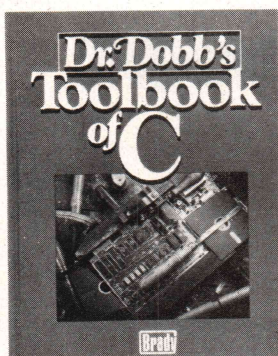
Small-Windows: A Library of Windowing Functions for the C Language

by James E. Hendrix

Small-Windows is a complete windowing library for C.

The package includes: 18 video functions written in assembly language, 7 menu functions that support both static and pop-up menus, 41 window functions to clean, frame, move, hide, show, scroll, push, and pop windows. Two test programs are provided as examples to show you how to use the library and the window, menu, and directory functions. Documentation and full C source code is included. Available for MS-DOS systems for the following compilers: Microsoft C Version 4.0, Small-C, Lattice C and Turbo C. Please specify compiler format.

Manual & Disk (MS-DOS)	Item #35-6	\$29.95
-----------------------------------	-------------------	----------------



Dr. Dobb's Toolbook of C

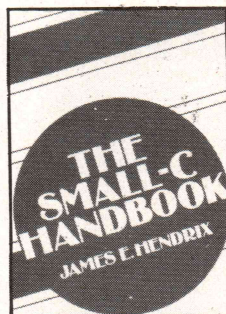
by the Editors of
*Dr. Dobb's Journal of
Software Tools*

This authoritative reference contains over 700 pages of the best C articles and source code from *Dr. Dobb's Journal of Software Tools*, along with new material by C experts. You'll find hundreds of pages of useful C source code, including a complete compiler, an assembler, and text-processing utilities.

Book

Item #615-3

\$29.95



The Small-C Compiler and Small-C Handbook

by James E. Hendrix

This compiler and handbook provide everything you need for learning how compilers are constructed, and for learning C at its most fundamental level. You'll find a discussion of assembly language concepts and program translation tools, and learn how to generate a new version of the compiler. Full source code is included. The handbook and compiler on disk are available for both MS-DOS and CP/M systems. The handbook comes with an addendum for the MS-DOS version. Please specify format.

Book & Disk (MS-DOS)
Book & Disk (CP/M)

Item #76-3
Item #67-4

\$42.90
\$37.90

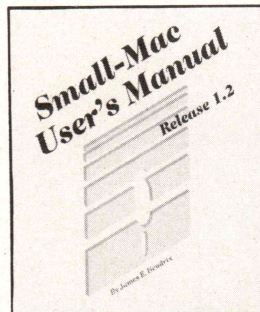


Small-Tools: Programs for Text Processing

by James E. Hendrix

This package of programs performs specific, modular operations on text files, including: editing; formatting; sorting; merging; listing; printing; searching; changing; transliterating; copying; concatenating; encrypting and decrypting; replacing spaces with tabs and tabs with spaces; counting characters, words, or lines; and selecting printer fonts. Small-Tools is supplied in source code form. With the Small-C Compiler, you can select and adapt these tools to your own purposes. Documentation is also included. Available for MS-DOS or CP/M systems. Please specify format.

Manual & Disk (MS-DOS or CP/M) **Item #78-X** **\$29.95**



Small-Mac: An Assembler for Small-C

by James E. Hendrix

This assembler features simplicity, portability, adaptability, and educational value. The package includes: a simplified macro facility; C language expression operators; object file visibility; descriptive error messages; an externally defined instruction table. You get the macro assembler, linkage editor, load-and-go loader, library manager, CPU configuration utility, and a utility to dump relocatable files. Documentation is included. For CP/M systems only. Please specify format.

Manual & Disk (CP/M)

Item #77-1

\$29.95

To Order:

Return the postage-paid Order Card on the next page, or
CALL TOLL FREE 800-533-4372 (In CA 800-356-2002)

Special Packages

Save 15%

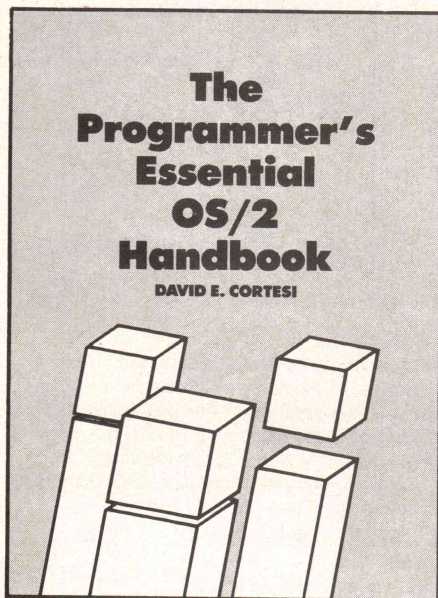
Receive the **C Chest** book & disk, the **Turbo C** book & disk, and the **Small-Windows** manual & disk, all for only \$93.95! You save 15%

C Chest/Turbo/Window Package **Item #168** **\$93.95**

Just Released!

Maximize the Power of OS/2

The Programmer's Essential OS/2 Handbook



by David E. Cortesi

The *Programmer's Essential OS/2 Handbook* will let you harness the power of OS/2, without getting lost in its intricacies. This hands-on, working reference book is organized to guide you through the complex features of the new OS/2 system. You'll even find detailed technical information that is not included in the official OS/2 documentation.

Multiple indices and a web of cross-referencing provide easy access to all OS/2 topic areas. Equal support for Pascal and C programmers is provided. You'll find:

- an overview of OS/2 architecture and vocabulary, including references to where the book handles each topic in depth
- a look at the 80286 and a description of how the CPU processes data in real and protect mode
- an overview of linking, multiprogramming, file access, and device drivers
- in depth discussions of important OS/2 topics, including dynamic linking; the message facility; the screen group; keyboard, mouse, and screen input, output and monitors; the queue; the semaphore; the thread; the process; storage allocation; segment swapping; and IOCTL usage
- detailed accounts of nearly 300 system calls, including DOS calls, keyboard calls, video calls, mouse calls, and device driver aides.

The Programmer's Essential OS/2 Handbook is a resource no programmer developing in the OS/2 environment can afford to be without.

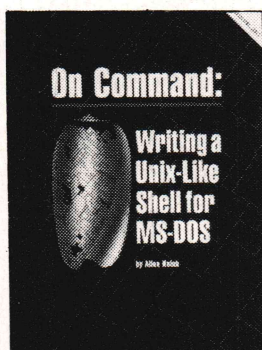
Book & Disk	Item #89-5	\$39.95
(PC/AT 5¼" 1.2 megabyte, or 3½" 2 megabyte)		
Book	Item #82-8	\$24.95

To Order:

Return the postage-paid Order Card, or
CALL TOLL FREE 800-533-4372 (In CA 800-356-2002)

Bring the Conveniences of UNIX To YOUR MS-DOS Machine

Full Source Code on Disk!



On Command: Writing a Unix-Like Shell for MS-DOS

by Allen Holub

This book and ready-to-use program demonstrate how to write a Unix-like shell for MS-DOS, with techniques applicable to most other programming languages as well. The book and disk include a detailed description and working version of the Shell, complete C source code, a thorough discussion of low-level MS-DOS interfacing, and significant examples of C programming at the system level.

Supported features include: read, aliases, history, redirection and pipes, Unix-like command syntax, MS-DOS-compatible prompt support, C-Shell-based shell scripts, and a Shell variable that expands to the contents of a file so a program can produce text that is used by Shell scripts. The Unix-like control flow includes: if/then/else; while; foreach; switch/case; break; continue.

The ready-to-use program and all C source code are included on disk. For IBM PC and direct compatibles.

Book & Disk (MS-DOS)

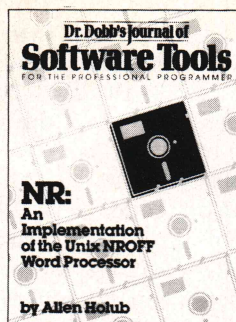
Item #29-1

\$39.95

Save 15%

Receive **On Command**, **Util** and **NR** together for only \$85.95!
You get all the convenience of Unix-like features and save 15%.

Unix-like Features Package **Item #167** **\$85.95**



NR: An Implementation of the Unix NROFF Word Processor

by Allen Holub

NR is a text formatter that is written in C and compatible with UNIX's NROFF. Complete source code is included in the **NR** package so that it can be easily customized to fit your needs. **NR** also includes an implementation of the -ms (manuscript) macro package and an in-depth description of how -ms works. **NR** does hyphenation and simple proportional spacing. It supports automatic table of contents and index generation, automatic footnotes and endnotes, italics, boldface, overstriking, underlining, and left and right margin adjustment. **NR** also contains:

- extensive macro and string capability
- number registers in various formats, including Roman and Arabic numerals, both spelled out and in outline form
- diversions and diversion traps (macros that are triggered automatically)
- input and output line traps

NR is easily configurable for most printers. Both the ready-to-use program and full source code are included.

For PC compatibles.

Manual & Disk (MS-DOS)

Item #33-X

\$29.95



\Util

by Allen Holub

When used with the Shell, this collection of utility programs and subroutines provides you with a fully functional subset of the Unix environment. Many of the utilities may also be used independently. You'll find executable versions of cat; cp; date; du; echo; grep; ls; mkdir; mv; p; pause; printevn; rm; rmdir; sub; and chmod.

The \Util package includes complete source code on disk and all programs (and most of the utility subroutines) are fully documented in a Unix-style manual. For IBM PC and direct compatibles.

Manual & Disk (MS-DOS)

Item #12-7

\$29.95

To Order:

Return the postage-paid Order Card, or

CALL TOLL FREE 800-533-4372 (In CA 800-356-2002)

size of their bounding rectangles. Figure 1, also below, shows some of the PICT images. Figure 2, page 100, shows the ICONs and ICN#s. I also removed the CDEF from the decompilation (more meaningless hex). It's produced by compiling rectCDEF.asm, as described later.

Lightspeed C takes the resources in custom controls demo PROJ.rsrc and binds them into the finished custom controls demo application. Other Mac development systems let you perform congruent conjunctions.

Custom Controls Demo's Resources

BNDL 128—This BuNDLe resource connects the program to its Finder icon.

CDEF 40—The Control Definition code.

CNTLs 1 thru 21—CoNTroL specifications for each of the program's main modal dialog's 21 custom control items.

CuCo 0—The application's version data/signature/creator resource: a Pascal-type string providing its name, version, and date.

DITL 1—Dialog ITeM List for the program's main modal dialog.

DITL 210—Dialog ITeM List for the

copyright button's modal dialog. DLOG 1—DiaLOG template for the program's main modal dialog. DLOG 210—DiaLOG template for the copyright button's modal dialog. FREF 128—File REference that hooks the program up with its Finder icon.

ICN# 128—The program's Finder icon.

ICONS 30, 110, 120, 121, 130, 140, 141, 150, 160, and 161—ICONS used with various control items in the main modal dialog. Numbering convention: the control's item number in the dialog times 10, with 1 added for a content-change image.

MENU 1—Menu template for the program's menu.

PICTs 50, 51, 60, 61, 70, 80, 90, 100, 101, 170, 171, 200, and 201—PICTures used with various control items in the main modal dialog. Numbering convention: the control's item number in the dialog times 10, with 1 added for a content-change image. PICT 210—PICTure used with the copyright button's modal dialog. STRs 20 and 40—STRings used for content changing with the modal dialog's control items 2 and 4.

PICTs 50 and 51—see last month's Figure 3, variation 5—rect: 0,0,57,61

PICTs 60 and 61—see last month's Figure 3, variation 7—rect: 0,0,68,69

PICT 70—see last month's Figures 4 and 7, DITL item 7
—rect: 0,0,34,32

PICT 80—see last month's Figures 4 and 7, DITL item 8
—rect: 0,3,196,116

PICT 90—see last month's Figures 4 and 7, DITL item 9
—rect: 0,12,35,73

PICTs 100 and 101—see last month's Figure 3, variation 9—rect: 0,0,56,70

PICTs 170 and 171—see last month's Figures 4 and 7, DITL item 17, and this month's Figure 2—rect: 0,0,187,85

PICTs 200 and 201—see last month's Figures 4 and 7, DITL item 20, and this month's Figure 2—rect: 0,0,24,24

PICT 210—see last month's Figure 2—rect: 1,14,168,290

Table 1: Finding the PICTs, and their bounding rectangles, from custom controls demo

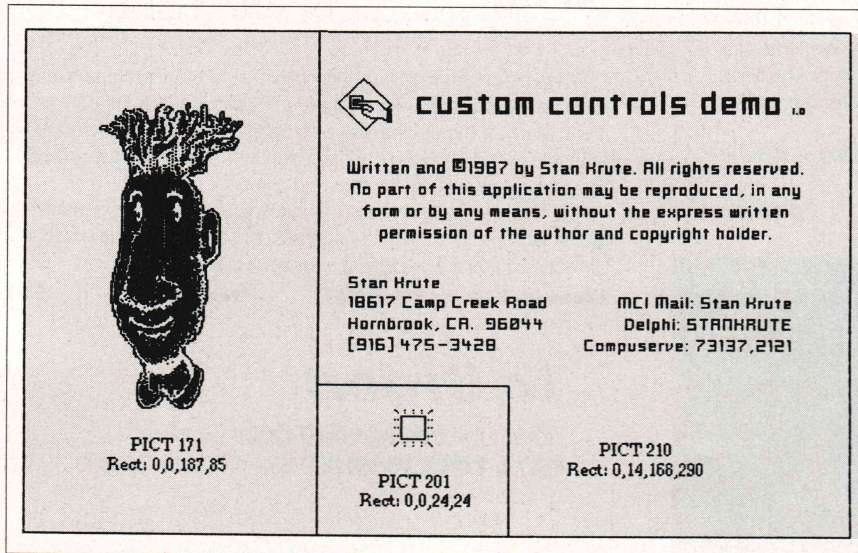


Figure 1: Selected PICTs from custom controls demo

CDEF Development Files

As mentioned in last column, the CDEF is written in assembly language, using MDS 2.1. Figure 3, page 100, shows the files involved in its development. rectCDEF.asm and rectCDEFequ.txt are the critical files; the others are either MDS-specific or adjuncts to development. Here are brief descriptions:

rectCDEF.job—An MDS executive file used to control one iteration of the development cycle. After compiling the CDEF, it turns it into a code resource, puts the resource into several files, and ends up in custom controls demo, ready for testing. See Listing Two, page 65.

rectCDEF.asm—Assembly-language source code for the CDEF. Entry point is at the beginning of the code. See Listing Three, page 66.

rectCDEFequ.txt—Private definitions for rectCDEF.asm. See Listing Four, page 83.

rectCDEF.link—Instructions for the MDS linker. Takes the .rel file produced by assembling rectCDEF.asm and produces a standard Mac CODE

Quit Wasting Time!

As a programmer, most of your time is spent writing and debugging source code, and documenting your work. A powerful, easy-to-use programmable text editor could be saving you HOURS of unnecessary effort.

Only MULTI-EDIT has all these time-saving features:

Fully automatic Windowing and Virtual Memory.

Edit multiple files regardless of physical memory size.
Easy cut-and-paste between files.
View different parts of the same file.

Powerful, EASY-TO-READ high-level macro language.

Standard language syntax.
Full access to ALL Editor functions.
Automate repetitive tasks.
Easy, automatic recording of keystrokes.

Language-specific macros for ALL major languages.

Smart indenting.
Smart brace/parenthesis/block checking.
Template editing.
Supports C, Pascal, BASIC and Assembler.

Terrific word-processing features for all your documentation needs.

Intelligent word-wrap.
Automatic pagination.
Full print formatting with justification, bold type, underlining and centering.
Even a table of contents generator.

Compile within the editor.

Automatically positions cursor at errors.
Built-in MAKE capabilities.
Run compiled program without leaving editor.
Automatically allocates all available memory to compiler or program.



Complete DOS Shell.

Scrollable directory listing.
Copy, Delete and Load multiple files with one command.
Background file printing.

Regular expression search and translate.

Condensed Mode display, for easy viewing of your program structure.

Pop-up FULL-FUNCTION Programmer's Calculator and ASCII chart.

**and MOST IMPORTANT,
the BEST user-interface on the market!**

- Extensive context-sensitive help.
- Choice of full menu system or logical function key layout.
- Function keys are always labeled on screen (no guessing required!).
- Excellent online, interactive tutorial.
- Keyboard may be easily reconfigured and re-labeled.

Users of Wordstar and Turbo Pascal's Editor could be programming in a fraction of the time with these features.

NO EDITOR ON THE MARKET TODAY HAS ALL THESE FEATURES, OR OFFERS YOU THIS MUCH POWER AT A REASONABLE PRICE, EXCEPT

Multi-Edit \$99. COMPLETE
VERSION 2.0 Or Get our FULLY FUNCTIONAL DEMO
Copy for only \$10!

	Multi-Edit	BRIEF 2.0	Norton Editor	Vedit Plus
Edit 20+ files larger than memory	Yes	Yes	No	Yes
Powerful high level macro language	Yes	Yes	No	Yes
Full UNDO	Yes	Yes	No	No
Visual marking of blocks	Yes	Yes	Yes	No
Column oriented block operations	Yes	Yes	No	No
Automatic file save	Yes	Yes	No	No
Online help	Extensive	Limited	Limited	Limited
Online tutorial	Yes	No	No	Yes
Choice of keystroke commands or menu system	Yes	No	No	Yes
Function Key assignments labeled on screen	Yes	No	No	No
WP Functions	Extensive	Limited	Limited	Extra Cost
Complete DOS shell	Yes	No	No	No
Pop-up Programmer's Calculator and ASCII table	Yes	No	No	No ASCII
Unlimited 'Off the Cuff' keystroke macros	Yes	No	No	Yes
Allocates all available memory to compiler when run from within editor	Yes	No	Yes	Yes
Intelligent indenting, template editing and brace/parenthesis/block matching and checking for all major languages	Yes	C Only	No	Limited
Flexible condensed mode display	Yes	No	Yes	No
Optional background communications and Spell Checker modules	Yes	No	No	No
PRICE	\$99	\$195	\$50	\$185

Requires IBM/PC/XT/AT/PS2 or full compatible, 256K RAM, PC/MS-DOS 2.0 or later. Multi-Edit and American Cybernetics are trademarks of American Cybernetics. BRIEF is a trademark of Underware, Inc. Norton Editor is a trademark of Peter Norton Computing, Inc. Vedit is a registered trademark of CompuView Products Inc. Copyright 1987 by American Cybernetics.

Get our FULLY FUNCTIONAL DEMO Copy for only \$10!

To Order, Call 24 hours a day:
1-800-221-9280 Ext. 951
In Arizona: 1-602-890-1166
Credit Card and COD orders accepted

American Cybernetics
138 Madrid Plaza
Mesa, AZ 85201

TO THE MACS

(continued from page 98)

resource. See Listing Five, page 83.

rectCDEF.R0—Instructions for the MDS resource compiler. Takes the standard CODE resource produced by the linker and turns it into a CDEF. See Listing Six, page 84.

rectCDEF.R1—Instructions for the MDS resource compiler. Takes the

CDEF produced via rectCDEF.R0 and adds it to custom controls demo. See Listing Seven, page 84.

rectCDEF.R2—Instructions for the MDS resource compiler. Takes the CDEF produced via rectCDEF.R0 and adds it to custom controls demo PROJ.rsrc. See Listing Eight, page 84.

rectCDEF.Rel—The .Rel file produced by assembling rectCDEF.Asm.

rectCDEF—The file produced by linking rectCDEF.Rel under the control of rectCDEF.Link.

rectCDEF.Rsrc—The file produced by the MDS resource compiler that contains the CDEF resource.

custom controls demo—My application, to which the MDS resource compiler adds the CDEF for easy testing during CDEF development.
custom controls demo PROJ.rsrc—My application's resources, to which the MDS resource compiler adds the CDEF.

CDEF Refresher

A C function prototype for a CDEF looks like this:

```
pascal long someCDEF ( int varCode,
ControlHandle theControl, int message,
long param ) ;
```

varCode indicates which of the CDEF's variations to use; *theControl* gives you a handle to the calling control's control record; and *message* tells the CDEF what to do. There are nine possible messages—see *Inside Macintosh* and/or the message jump table in the source code. Finally, *param* is 4 byte of data that vary according to the message passed to the function. The meaning of the function result also varies with the message; the default return value is 0.

The *pascal* keyword in the prototype indicates that the function is called with Pascal calling conventions. In rectCDEF.Asm, I use LINK/UNLK to create space for some local variables. Figure 4, page 101, shows how the stack looks after the CDEF is called and the LINK A6 command's been given.

The Clean Nut Can't Shut Up

As I never tire of pointing out: good Mac programming has to play by the rules. Keep it clean, and your programs will run now and in the future. What entaileth clean? Deal with error codes. Lock down heap objects only when you must, and let them float whenever you can. Assume nothing. Steer clear of low memory. Follow *Inside Mac* and Apple's *Mac Tech Notes* religiously. Etc., etc., etc.

Gawd, I sound like the Ms. Grundy of cybernetics. But, hey,

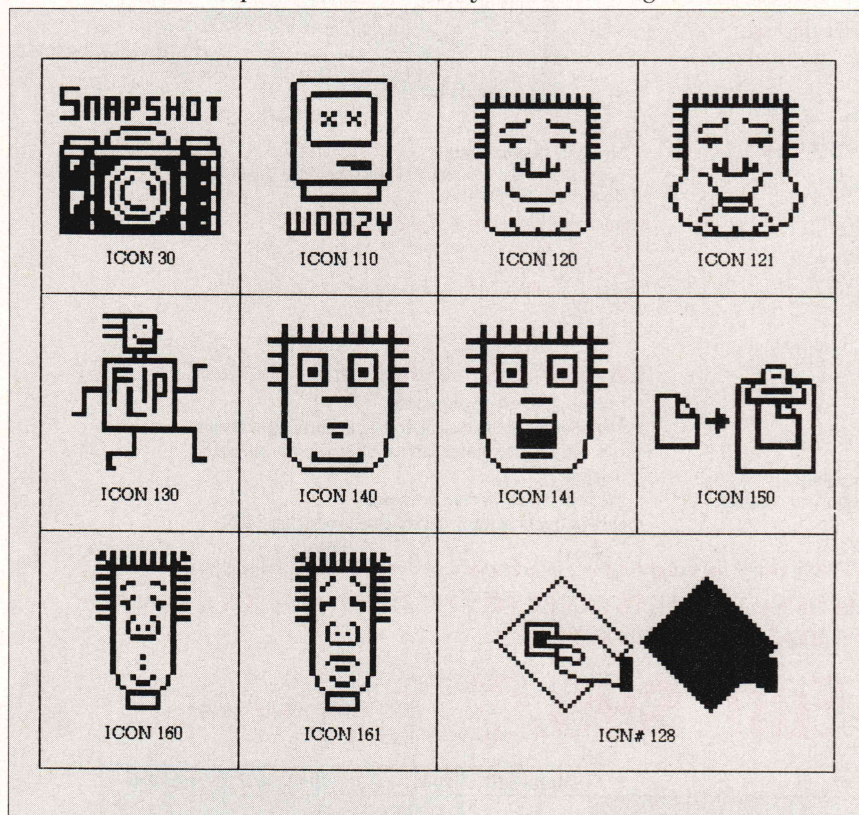


Figure 2: ICONs and ICN#s from custom controls demo

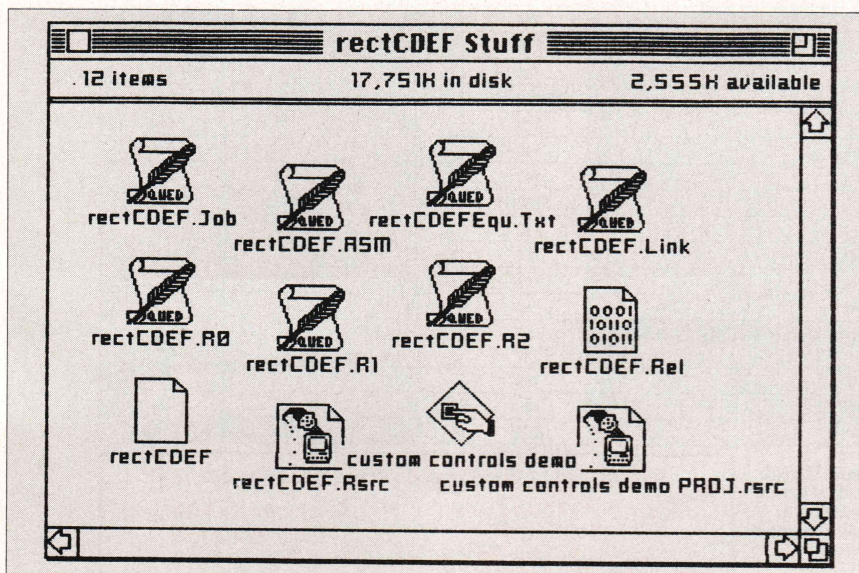


Figure 3: The rectCDEF files

cleanliness works. Clean Mac programs I wrote three years ago function on Mac IIs under MultiFinder.

Take a look at the *rectCDEF* code, for example. The control that's calling the CDEF gets its control record locked down during any call to the CDEF (see the CDEF's entry area). The same with the control's auxiliary data block (also in the CDEF's entry area). Both of these data objects are released at the end of the CDEF call. Calls to the resource and memory managers (in *doInitCntrl*) are carefully checked for success/failure, and the program tries to respond reasonably to the results (see *doInitCntrl* and *doDispCntrl*).

This is a 680x0 we've got, with all those lovely registers waiting to serve. So pack 'em up. That's what I try to do in the CDEF. Here's the scoop: Three address registers are taken up right off the bat. A7 points to the stack, A6 points to a stack frame, and A5 fingers application and Quickdraw globals. Then, I use A2 to point to the control's record and A3 to point to its auxiliary data block. That leaves A4 free for temporary usage and A0 and A1 for even more temporary usage. That's because ROM/OS calls generally trash A0 and A1.

Data registers next: I use D3 to hold the function parameter *param*. D4 holds flags that let me quickly scoot around the code based on which of the 16 CDEF variations is in effect. That leaves D5, D6, and D7 free for temporary usage and D0, D1, and D2 for even more temporary usage. Again, that's because ROM/OS calls can trash D0, D1, and D2.

On entry to the CDEF, I build a stack frame, save registers that I'll use, grab some parameters, grab a set of variation flags, set a default function result, lock the control record down, point to any auxiliary data block and lock it down (if it exists), then jump to a specific routine based on the function's message parameter.

Returning from that specific routine, I make a somewhat symmetric exit: unlock any auxiliary data block, unlock the control record, restore saved registers, remove the stack frame, and jump on back to the CDEF's caller.

The CDEF handles 16 control vari-

ations. I use an 8-bit flag to signal eight qualities of a given control variation. This simplifies the rest of the CDEF function's control logic. Take a look at *varFlagTable* for details.

Handling Controls

doInitCntrl takes care of any special initialization needs. For this CDEF, I need an auxiliary data block to hold handles to variant-specific resources.

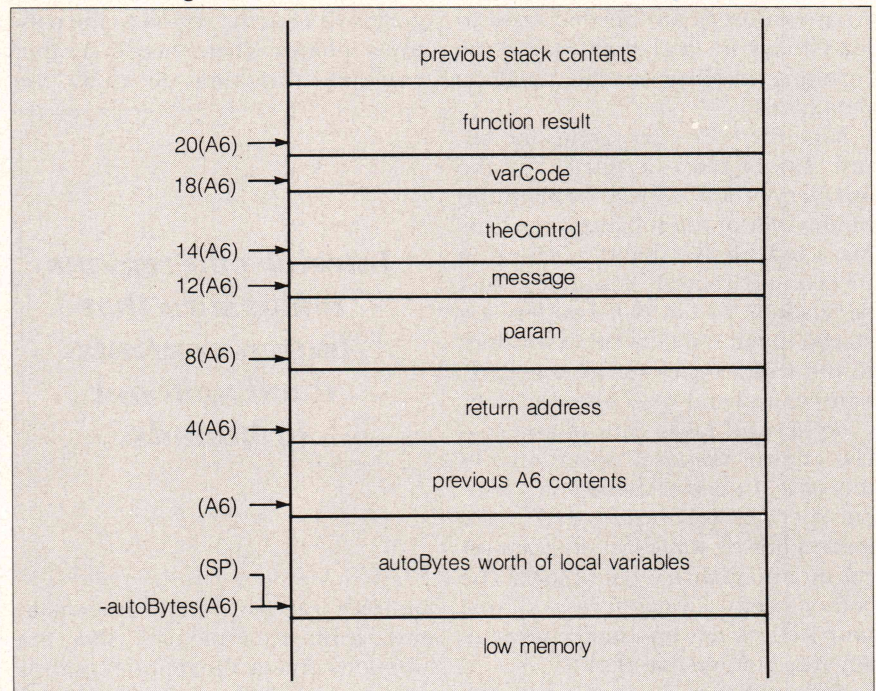
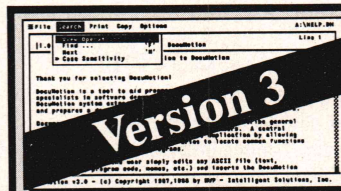


Figure 4: The state of the stack upon entry to *rectCDEFProc*, just after the LINK instruction



Documentation is a PAIN!

... without **DocuMotion™**

We've found that well indexed and easily accessed documentation is a powerful tool. Now you can simply pop up **DocuMotion** to access, display and print *your* documentation or guide reference libraries. **DocuMotion** builds cross indexed document libraries from documentation contained in your source code or any text file. **Version 3** adds great features like: folding documents, cross-referencing, 10 book marks, and a context-sensitive help interface.

DocuMotion is PAINLESS
IT'S DOWN-RIGHT FUN!

... for Programmers:

- Your documentation is available ANYWHERE, ANY TIME.
- Access, display and print your documentation by name or by user-defined categorization trees.
- Your choice -- pull-down menus or intuitive accelerator keys.
- Powerful print & copy functions.

... for Project Managers:

- Programmers produce more and better documentation.
- Reduced function redundancy.
- Greater programmer productivity.

... for the PC:

- Runs memory resident or non-resident on any IBM PC/XT/AT or compatible.
- Compatible with all popular memory resident programs.
- LAN compatible.

DocuMotion is for YOU:

Call NOW 1-612-884-5860

Developer's System **\$159.95**

Coming Soon: Reference Guides for your favorite tools...CALL.

We need your expertise for reference guides. Any Topic. Liberal royalties...CALL

NW - Intelligent Solutions, Inc.

P.O. Box 20478 Bloomington, MN. 55420-0478

So *doInitCntl* first tries to get a block. If it succeeds, it then looks to see what resources are needed, tries to load them in, and then stores the resulting handles in the auxiliary data block.

Note the code's provisions for failure. No memory available for the auxiliary data block? You jump nimbly out of the initialization, leaving a tell-tale *NIL* handle behind in the control record. Resources can't be loaded? Again, *NIL* handles are stored to tell the tale. Other routines in the CDEF check for all these *NIL* handles and and react appropriately.

doDispCntl takes care of any special control disposal operations. In this case, I need to release any variant-specific resources that were loaded during initialization, then get rid of any auxiliary data block. The code's pretty straightforward. Note how I check for, and dance around, any *NIL* pointers/handles.

doDrawCntl is invoked when the CDEF's asked to draw the control. It

first saves grafport features it may change. Then it cases out on the type of outline the control needs: shadowed, simple, or none. After drawing the outline, it sets the grafport's clipping region to the control's interior. Then there's another case-out, this time to draw the

***Interior clip regions
make sure that
button contents
don't spill out
of bounds.***

button's interior, which can contain text, icon, or a picture. After the interior's drawn, the routine restores the saved grafport features and ends.

Outlines and Interior Clip Regions

It's fairly simple stuff, really. *rectCDEF* supports two styles of outlining: shadowed and simple. *doShadOutline* draws shadowed outlines. First it draws two shadow lines, then a rectangle. *doSimpOutline* draws simple outlines by drawing a rectangle.

Interior clip regions make sure that button contents don't spill out of bounds. For outlined buttons, you just take a rectangle slightly inset from the button's bounding rectangle. Naked buttons use the bounding rectangle.

Drawing Button Interiors

doTextInterior draws text button interiors. It starts by figuring out which font the button's text will get drawn in and sets the grafport accordingly. Then it uses the font information to figure out vertical positioning for the text.

Next, the routine gets a pointer to the text string it'll draw. This can be the control's title or, in the case of a content-changing button, a *STR* re-

**New Titles for the
Computer Professional
from Springer-Verlag**

New

Software Engineering in C

P.A. Darnell and P.E. Margolis

"... *Software Engineering in C* is much more than how to write legal C, providing pointers on how to write clear efficient C and what pitfalls to watch out for on the way."

— Bob Scheulen

Senior Software Engineer, Microsoft®

The book includes C's more advanced features, a chapter on software engineering, a listing of differences between the ANSI and K&R Standards, and a C Interpreter Listing.
1988/612 pp/62 illus/59 tables/Paper \$29.95
ISBN 0-387-96574-2

(Springer Books on Professional Computing)



Springer-Verlag

New York Berlin Heidelberg Vienna
London Paris Tokyo

New

An Introduction to TCP/IP

J. Davidson

This book describes TCP/IP, the most important emerging internet protocol suite in the computer networking field. Material is presented according to the Seven Layers of the ISO Reference Model, each Layer of which provides a specified level of networking service. The book should be a valuable source of information for those interested in local or wide-area computer networking projects.

1988/100 pp./30 illus./Softcover \$24.95

ISBN 0-387-96651-X

New

**The Unix System Guidebook
Second Edition**

P.P. Silvester

This book presents the Unix system for the beginning user. While the emphasis is on System V, the book also makes note of Berkeley 4.2 BSD and Seventh Edition Unix. Major features of this second edition include in-depth coverage of editing with *vi* and formatting text with *nroff*, an increased discussion of Shell programming, and a section on using Writer's Workbench.

1988/325 pp/15 illus./Softcover \$35 (tent.)

ISBN 0-387-96489-4

(Springer Books on Professional Computing)

Taming the Tiger

Software Engineering and Software Economics

L.S. Levy of AT&T Bell Laboratories

The book presents a new approach to software development based on the analytical techniques of managerial economics. Models presented in the book show how to apportion development costs among different phases of the development cycle. The book should be of special interest to software engineers and managers of software development.

1987/248 pp/9 illus./Softcover \$25.00

ISBN 0-387-96468-1

(Springer Books on Professional Computing)

To order by credit card, please call our toll-free number: 1-800-526-7254 (in NJ call 201-348-4033). To pay by check (include \$2.50 for postage and handling; N.Y., NJ, and CA residents please add sales tax) or for more information, write to:

Springer-Verlag New York, Inc.
Computer Science Promotion
175 Fifth Avenue
New York, N.Y. 10010

SUDDENLY, MAGIC PC MAKES YOUR DBMS OBSOLETE

Attn
Btrieve
programmers:

You know how database applications are created — by hacking out line after line of time-consuming code. Most DBMS' and 4GL's give you some programming power. But when it comes to serious applications, they keep you bolted to your seat writing mountains of tedious code. And rewriting it all over again with every design change.

Imagine how much faster you'd be if you could replace the painful coding phase with an innovative visual technology which takes only a fraction of the time: Introducing Magic PC—the revolutionary Visual Database Language from Aker Corporation:

• High-Speed Programming:

With Magic PC's visual design language you quickly describe your programs in non-procedural Execution Tables. They contain compact programming operations which are executed by Magic PC's runtime engine. You fill-in the tables using a visual interface driven by windows and point-and-shoot menus. One table with 50 operations eliminates writing more than 500 traditional lines of code. Yet with Magic PC you don't sacrifice any power or flexibility.

With a powerful set of high-level non-procedural operations you program at only a fraction of the time.

• Maximum Power AND Simplicity:

With Magic PC, you can generate robust DBMS applications including screens, windows, menus, reports, forms, import/export, and much more! Plus, Magic PC has one of the friendliest user interfaces you've ever seen. Using Magic PC you can look-up and transfer data through a powerful Zoom Window system. Magic PC even lets you perform command-free queries.

• Btrieve Performance:

Magic PC incorporates Btrieve, the high-performance file manager from SoftCraft. This gives you exceptional access speed, extended data dictionary capabilities, and automatic file recovery!

• Virtually Maintenance-Free:

With Magic PC you can modify your application design "on the fly" without any manual maintenance. Magic PC automatically updates your programs and data files on-line! This also makes Magic PC an ideal tool for prototyping complete applications in hours instead of days.

• FREE Networking:

Magic PC comes complete with LAN features. Develop multi-user applications for your LAN with Magic's file and record-locking security levels.

• Stand-Along Runtime:

Distribute your applications and protect your design with Magic PC's low cost runtime engine.

• All For Only \$199:

Best of all, Magic PC is an unbeatable bargain. For a limited time, Magic PC's price has been reduced to only \$199! Yes, this is the same Magic PC that normally lists for \$695! And Magic PC eliminates the need for a separate DBMS, compiler, or application generator. It comes complete with all the tools you need to develop your own database applications instantly.



"Magic PC's data base engine delivers powerful applications in a fraction of the time... there is truly no competitive product."

Victor Wright — PC Tech Journal

Pop-up Zoom Windows run multiple programs per screen — with point-and-shoot data transfer between windows!

• \$199 — With A Money-Back Guarantee!

For a limited time, you can get Magic PC for only \$199. And even at this low price, Magic PC is risk-free. If you're not completely satisfied, simply return it within 30 days and we'll buy it back (less \$19.95 restocking fee). And if you'd like a preview, Magic PC's Tutorial Demo is available for just \$19.95. But you'd better hurry — Magic PC's special \$199 price won't last long!

• Join The Magic PC Revolution

To unleash your DBMS design power, order your \$199 copy of Magic PC right now by calling toll-free or returning the coupon below.

ORDER NOW: CALL
(800) 345-MAGIC
In CA (714) 250-1718

MAGIC PC
The Visual Database Language
by **AKER**



Yes! I want to generate powerful applications much faster!

☐ Rush me my copy of Magic PC at the special promotional price of \$199 (add \$10 P&H, and tax in CA. International orders add \$30). I understand I can return Magic PC for a refund within 30 days, if I'm not completely satisfied.*

☐ Rush me a copy of Magic PC Tutorial Demo at \$19.95 (add \$5 P&H, and tax in CA. International orders add \$15).

Name _____
Company _____ Phone _____
Street Address (no POB) _____
City _____ State _____ Zip _____
☐ Check enclosed ☐ Charge to my: ☐ VISA ☐ MC ☐ AMERICAN EXPRESS
Account No. _____
Acct. Name _____ Exp. Date _____
Signature _____
Return to: **Aker Corp., 18007 Skypark Cir B2, Irvine, CA 92714**

System requirements: IBM PC, XT, AT, PS/2 or 100% compatible with 512K RAM, hard disk and DOS 2.0 or later. 514" format, not copy protected. Dealer pricing available. *Return policy valid in US only.

Aker, Magic PC, The Visual Database Language are trademarks of Aker Corporation. All other trademarks acknowledged. © Copyright 1987, Aker Corp.



DBMS Pros:
TRY MAGIC PC -
THE ULTIMATE
POWER & SPEED
BEYOND CODING
\$695
Now \$199.

TO THE MACS

(continued from page 102)

source. If it's a STR resource, the resource gets locked down. The routine uses the text string to figure out horizontal positioning. Now that the vertical and horizontal positions are known, *doTextInterior* moves the grafport's drawing pen into place to start drawing the string.

Depending on the button's type and state, the interior is painted white or black. Based on the same information, the routine draws the text string in black or white. If a STR resource supplied the text, it's now unlocked. And, if the button's in an inactive state, black text gets turned to gray. The routine restores the grafport and exits.

doPictInterior draws picture button interiors. First it figures out the PICT resource to use, then it locks the PICT down. It uses the PICT's height and width to set up a destination rectangle that'll center the picture in the button's interior. If the picture's too big, the destination rectangle is set so its upper-left

corner matches the button interior's upper-left corner. Then the routine erases the button's interior and draws the PICT, and the PICT gets unlocked. Finally, a separate routine, *interiorAdjust*, is called to deal with any necessary image inverting or graying out.

Very similarly, *doIconInterior* draws iconic button interiors. It uses the standard height and width of an icon to set up a destination rectangle that'll center the icon in the button's interior. If the icon's too big, the destination rectangle is set so its upper-left corner matches the button interior's upper-left corner. Then the routine erases the button's interior and draws the icon. Finally, *interiorAdjust* is called to deal with any necessary image inverting or graying out.

Procedural wrap up

doTestCntl tests a point to see if it's contained within an active control. If it is, the routine sets the CDEF's function result to the control's part number. If the point isn't in the control, or if the control's inactive,

the routine sets an appropriate result code.

Because my controls are rectangular, the containment test is simple: just call on the Mac's *PtInRect* function, which checks for a point's containment within a rectangle.

DoCalcCCntl is deliciously trivial. Somebody wants the control described as a region? You just send the control's rectangle to the Mac's *RectRgn* procedure, which takes a rectangle and produces a corresponding region description.

There are four messages to the CDEF that you ignore, relying instead on the Mac's default behaviors. So *doPosCntl*, *doThumbCntl*, *doDragCntl*, and *doAutoTrack* are just stubs.

Availability

All the source code for articles in this issue is available on a single disk. To order, send \$14.95 to Dr. Dobb's Journal, 501 Galveston Dr., Redwood City, CA 94063, or call (415) 366-3600, ext. 216. Please specify the issue number and format (MS-DOS, Macintosh, Kaypro).

C Programmers: Combine C and COMMON LISP to Increase the Power of Your Software

TransLISP PLUS™

Simple.

Add LISP features to your software without making it a full time job. The TransLISP PLUS tutorial, on-line help, and 30 sample programs with commented source make it easy.

Practical.

Start by modifying the LISP sample programs and including them in a system you wrote in C. Yes, in C! TransLISP PLUS includes a C Language Interface that lets you integrate your Microsoft C code and libraries with all or portions of our LISP interpreter.

Use TransLISP PLUS to add natural or command language features to replace menus... or to flexibly manage related but disparate information. Code from C libraries provided by other vendors can be integrated into your program to perform tasks not normally part of LISP.

Thorough.

TransLISP PLUS took over 400 primitives from the most widely used and respected LISP standard, COMMON LISP, and made it available on IBM PCs, XT's, AT's, and virtually every other MSDOS machine. So now you can work with anything from a \$700 PC to a \$7000 PC.

The utilities toolbox is included at no charge with a built-in editor, pretty printer, cross reference, and additional debugging tools.

An optional Runtime encrypts your source code so that you can distribute your applications safely. You pay no royalties.

Requires MSDOS 2.0+, 320K RAM, and a 360K floppy.

MONEYBACK GUARANTEE

Try TransLISP PLUS (\$195) for 30 days — if not satisfied get a full product refund. The Optional Runtime is available for \$150. Or start by learning LISP with TransLISP (\$95) then upgrade to PLUS for \$158.

Call (800) 255-4659

In MA (617) 331-0800



**The
Coder's
Source™**

541-D Main St., Suite 412, So. Weymouth, MA 02190

CIRCLE NO. 175 ON READER SERVICE CARD

Bibliography

Apple Computer Inc. Human Interface Guidelines: The Apple Desktop Interface. Human Interface and Technical Publications Groups APDA #KNBHIG.

Goodman, Danny. The Complete HyperCard Handbook. : Bantam, 1987. MultiFinder Development Package Version 1.0. Available from APDA #KMSMSD, \$12.50.

DDJ

(Listings begin on page 64.)

Vote for your favorite feature/article.
Circle Reader Service **No. 4.**

Vendors

APDA

(Apple Programmer's and Developer's Assoc.)
290 S.W. 43rd St.
Renton, WA 98055
(800) 426-3667
In Wash. (800) 527-7562
In Canada (800) 237-4644
(206) 251-5222

HyperCard

Available from Apple dealers

HyperCard Technical Reference Package

APDA #KMBHTL
Available from APDA
(see above)

MultiFinder 1.0

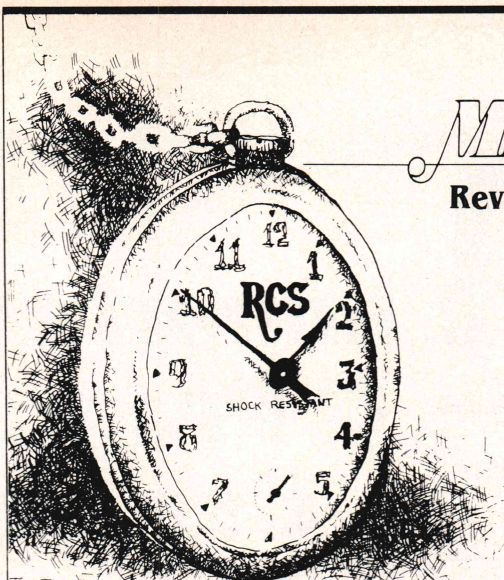
(plus System 4.2 and Finder 6.0)
Available from Apple dealers

ResTools 2.01

Alan Dahlbom
Downloadable from Delphi,
GENie, and other on-line services

Stack Detective

Peter Olson
Shareware available from Delphi
and other shareware sources



MKS RCS

Revision Control System

*Time
is on
my side*

With MKS **RCS** running under DOS, programmers, systems administrators, project managers and librarians can efficiently control and record the revisions of text files such as programs, documentation, graphs, papers, form letters, and so on. This software package:

- resolves access conflicts — for example, it prevents two programmers or authors from making simultaneous changes to a file;
- maintains a complete history of changes, including date and time of change, author, and reason for the change;
- allows retrieval of any version of the file, by date, release number, or a user-assigned name;
- runs quickly because only the most recent version of a file is stored — non-current versions are stored as difference-files to minimize storage requirements;
- allows divergent versions to branch from the main family of versions — these might be test versions of programs, or customized versions of documents;

Programs included in the package:

- | | |
|--|---|
| • ci — check in RCS revisions | • rcsmerge — merge RCS revisions |
| • co — check out RCS revisions | • rlog — display log messages about RCS files |
| • ident — display RCS identification information | • diff — show minimal file differences |
| • merge — three-way file merge | • diff3 — show differences among three files |
| • rfs — change RCS file attributes | |
| • rcsclean — clean up work files | |
| • rcsdiff — compare RCS revisions | |

The entire system for \$189.

Also available:

The MKS Toolkit: over 110 UNIX-based tools for DOS including the **Korn Shell**, **Vi**, and **AWK**, complete with nearly 400 pages of documentation and tutorials. The complete package: **\$139.**

MKS Vi: The UNIX screen editor running under DOS at lightning fast speeds — it's tuned for the PC. Comes with Tutorial and Reference Manual for **\$75.**

AWK: The 4th generation language written for DOS and fully compatible with the latest Bell Labs' specs: **\$75.** **AWK** and *The AWK Programming Language* by Aho, Weinberger, and Kernighan: **\$89.**

Mortice Kern Systems Inc.,

35 King Street North, Waterloo, Ontario, Canada N2J 2W9

(519) 884-2251

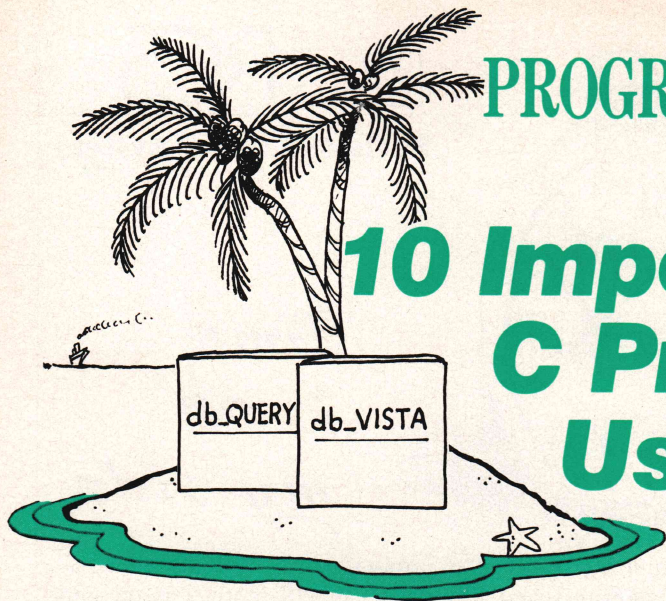
UUCP: ...!uunet!watmath!mks!inquiry

BIX User Name: mks

CompuServe User ID: 73260,1043

MKS software runs under MS-DOS 2.0 or later. Not copy protected. Prices quoted in US funds. VISA, MASTERCARD, American Express, uucp, and purchase orders (over \$200) are accepted. Overseas orders please add \$15 for postage and handling. MKS is a registered trademark of Mortice Kern Systems Inc. UNIX is a trademark of AT&T Bell Labs. MS-DOS is a trademark of Microsoft Corp.

CIRCLE NO. 176 ON READER SERVICE CARD



PROGRAMMER'S PARADISE PRESENTS

10 Important Reasons C Programmers Use db_Vista

from **RAIMA**TM
CORPORATION

RAIMA & Programmer's Paradise offer you the best value on the tools you need.

1. It's written in C.

Clearly the growing language of choice for applications that are fast, portable and efficient.

2. It's fast — almost 3 times faster than a leading competitor.

Fast access that comes from the unique combination of the B-tree indexing method and the "network" or direct "set" relationships between records.

3. It's flexible.

Because of db_VISTA's combination of access methods, you can program to your application needs with ultimate design flexibility. Use db_VISTA as an ISAM file manager or to design database applications. You decide how to optimize run-time performance. No other tool gives you this flexibility without sacrificing performance.

4. It's portable.

db_VISTA operates on most popular computers and operating systems like UNIX, MS-DOS and VMS. You can write applications for micros, minis, or even mainframes.

5. Complete Source Code available.

The entire C Source Code is available so you can optimize performance or port to new environments yourself.

6. It uses space efficiently.

db_VISTA lets you precisely define relationships to minimize redundant data. It is non-RAM resident; only those functions necessary for operation become part of the run-time program.

7. Royalty free run-time.

Whether you're developing applications for yourself or for thousands, you pay for db_VISTA or db_QUERY only once. If you currently pay royalties to someone else for your hard work, isn't it time you switched to royalty-free db_VISTA?

8. SQL-based db_QUERYTM

Add Raima's new C-linkable, SQL-based, ad hoc query and report-writing companion product to provide a simple relational view of your db_VISTA applications.

9. Free tech support.

60 days of free technical and application development support for every Raima product.

10. Upward database compatibility

Start out with file management in a single-user PC environment — then move up to a multi-user LAN or a VAX database application with millions of records.

db_VISTATM Features

- ◆ Multi-user support allows flexibility to run on local area networks
- ◆ File structure is based on the B-tree indexing method
- ◆ Transaction processing assures multi-user consistency
- ◆ File locking support provides read and write locks
- ◆ SQL-based db_QUERY is linkable
- ◆ File transfer utilities included for ASCII, dBASE optional
- ◆ Royalty-free run-time distribution
- ◆ Source Code available
- ◆ Data Definition Language for specifying the content and organization of your files
- ◆ Interactive database access utility
- ◆ Database consistency check utility
- File Management Record and File Sizes**
 - ◆ Maximum record length limited only by accessible RAM
 - ◆ Maximum records per file is 16,777,215
 - ◆ Maximum file size limited only by available disk storage
 - ◆ Maximum of 256 index and data files
 - ◆ Key length maximum 246 bytes
 - ◆ No limit on number of key fields per record
 - ◆ No limit on maximum number of fields per record
- Operating System & Compiler Support**
 - ◆ Operating systems: MS-DOS, PC-DOS, UNIX, XENIX, UNOS, ULTRIX, Microport, VMS
 - ◆ C compilers: Lattice, Microsoft, IBM, DeSmet, Aztec, Computer Innovations, Turbo C, XENIX and UNIX

db_VISTA or db_QUERY	List
Single user	\$195
Single user w/Source	\$495
Multi-user	\$495
Multi-user w/Source	\$990

Order Now

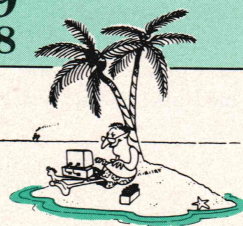
db_VISTA from Raima Corporation, put it to work in your application program.

Call Toll-Free Today!

1-800-445-7899
In NY: 914-332-4548

**Programmer's
ParadiseTM**

A Division of Hudson Technologies, Inc.
42 River Street, Tarrytown, NY 10591



Call or write
for the latest catalog



DISCOVER PARADISE

Programmer's Paradise Gives You Superb Selection, Personal Service and Unbeatable Prices!

Welcome to Paradise. The microcomputer software source that caters to your programming needs. Discover the Many Advantages of Paradise...

- Lowest price guaranteed
- Latest versions
- Huge inventory, immediate shipment
- Knowledgeable sales staff
- Special orders
- 30-day money-back guarantee

Over 500 brand-name products in stock—if you don't see it, call!

We'll Match Any Nationally Advertised Price.

	LIST OURS
386 SOFTWARE	
ADVANTAGE 386 C OR PASCAL	895 829
DESQVIEW	NEW 130 109
MICROPORT SYSTEM	
V/386 (COMPLETE)	SPECIAL 799 669
MS WINDOWS/386	SPECIAL 195 125
PHARLAP 386/ASM/LINK	495 419
SCO XENIX SYS V 386 (COMPLETE)	1495 1195
VM/386	245 179
386-TO-THE-MAX	NEW 75 65

ARTIFICIAL INTELLIGENCE	
ARITY STANDARD PROLOG	95 79
MULISP-87 INTERPRETER	300 199
PC SCHEME	95 85
SMALLTALK/V	SPECIAL 99 79
TURBO PROLOG	100 69
TURBO PROLOG TOOLBOX	100 69

ASSEMBLERS/LINKERS	
ADVANTAGE DISASM	SPECIAL 295 269
MS MACRO ASSEMBLER	150 99
OPTASM	SPECIAL, NEW 195 165
PASM86	195 115
PLINK86PLUS	495 279
RELMS, UNIWARE X-ASMS	CALL CALL

BASIC	
DB/LIB	139 119
FINALLY!	89 89
FLASH-UP	89 79
MACH2	75 59
MS QUICKBASIC	SPECIAL 99 65
QUICKPAK	69 59
QUICKWINDOWS	99 89
TRUE BASIC	100 79
TURBO BASIC	100 69
DATABASE TOOLBOX	100 69
EDITOR TOOLBOX	100 69
TELECOM TOOLBOX	100 69

MOUSE PRODUCTS

LOGITECH SERIAL OR BUS MOUSE	
W/PLUS, SOFTWARE	119 99
W/PLUS, LOGICPAINT	149 119
W/PLUS, LOGICADD MOUSE	189 149
W/PLUS, PUBLISHER MOUSE	179 145
W/PLUS, PAINT, CADD	209 169
W/PLUS, CADD PUBL. MOUSE	239 189
W/PLUS, PAINT, CADD, PUBL.	253 205
LOGITECH SERIES 2 W/PLUS	99 79
MICROSOFT SER OR BUS MOUSE	150 99
W/EASY CAD	175 119
W/MS WINDOWS	200 139
PC MOUSE BUS W/PNT & POPUPS	179 129
PC MOUSE SER. W/PNT & POPUPS	159 115
SUMMAMOUSE	119 99

C COMPILERS	
C86PLUS	497 375
LATTICE C	500 269
MICROSOFT C	450 285
QUICK C	99 65
TURBO C	100 65

C++	
ADVANTAGE C++	SPECIAL 495 469
PFORCE++	SPECIAL 395 199

C INTERPRETERS	
C-TERP	298 229
INSTANT C	495 379
RUN/C	120 79
RUN/C PROFESSIONAL	250 155

C LIBRARIES	
BASIC C	175 139
C ASYNCH MANAGER	175 135
C-FOOD SMORGASBORD	150 95
W/SOURCE CODE	300 179
C TOOLS PLUS/5.0	129 99
C UTILITY LIBRARY	185 125
C-XPERT	395 339
ESSENTIAL COMMUNICATIONS	185 125
COMMUNICATIONS PLUS	250 195
GREENLEAF C SAMPLER	95 69
GREENLEAF COMM LIBRARY	185 125
GREENLEAF FUNCTIONS	185 125
MULTI-C	149 95
PFORCE	295 215
RESIDENT C W/SOURCE	198 169
TIME SLICER	295 279
W/SOURCE CODE	1000 895
TURBO C TOOLS	129 99

COBOL	
E-Z PAGE	295 259
MICRO FOCUS	
COBOL/2	900 729
COBOL/2 TLSET	NEW 900 729
PC-CICS	1500 1000
LEVEL II COBOL	349 279
PERSONAL COBOL	149 119
OTHERS	CALL CALL
MICROSOFT COBOL	700 449
MICROSOFT SORT	195 129
OPT-TECH SORT	149 105
REALCICS	995 785
REALIA COBOL	995 785
W/REALMENU	1145 899
RM/COBOL	950 759
RM/COBOL-85	1250 999
RM/SCREENS	395 315
SCREENIO	400 379

DEBUGGERS	
ADVANCED TRACE-86	175 119
C-SPRITE	175 119
PERISCOPE I	345 275
PERISCOPE II	175 139
PERISCOPE III 8 MHZ	995 799
PERISCOPE III 10 MHZ	1095 875
PFIX 86 PLUS	395 215
T-DEBUG PLUS	60 49

DISK/DOS/KEYBOARD UTILITIES	
COMMAND PLUS	NEW V.2.0 80 69
DISK OPTIMIZER	60 55
FETCH	55 45
INTELLIGENT BACKUP	150 135
NORTON COMMANDER	75 55
ADVANCED NORTON UTILITIES	150 99
PDISK	145 105
VFEATURE	80 75

EDITORS	
BRIEF	195 CALL
WDBRIEF	275 CALL
EMACS	295 265
EPSILON	195 149
KEDIT	125 99

PC/EDIT	250 229
PEDITOR	195 155
PMATE	195 115
SPF/PC	195 145
VEDIT PLUS	185 129

FILE MANAGEMENT	
BTRIEVE	245 185
XTRIEVE	245 185
REPORT OPTION	145 99
BTRIEVE/N	595 455
XTRIEVE/N	595 455
REPORT OPTION/N	345 269
CBTREE	159 139
C-TREE	395 315
R-TREE	295 239
C-TREE/R-TREE BUNDLE	650 519
DBC III PLUS	250 169
DB VISTA OR DB QUERY	SPECIAL 195 159
SINGLE USER/SOURCE	SPECIAL 495 399
MULTIUSER	SPECIAL 495 399
MULTIUSER W/SOURCE	SPECIAL 990 789
INFORMIX PRODUCTS	CALL CALL
PHACT MANAGER	249 795
XQL	599 599

FORTRAN COMPILERS	
LAHEY FORTRAN S77L	SPECIAL 695 CALL
LAHEY PERSONAL FORTRAN 77	95 89
MICROSOFT FORTRAN	450 285
RM/FORTRAN	595 479

FORTRAN UTILITIES/LIBRARIES	
AUTOMATED PROGRAMMER	995 949
DIAGRAM DOCUMENTER	129 115
GRAMMATIC OR PLOTMATIC	135 119
MAGUS NUMERICAL ANALYST	295 249
MATHPAC	495 445
NO LIMIT	129 109
SPINDRIFT LIBRARY	149 135
SSP/PC	350 269

GRAPHICS	
ADVANTAGE GRAPHICS (C)	250 229
ESSENTIAL GRAPHICS	250 189
GSS GRAPHIC DEV. TOOLKIT	495 375
HALO	209 209
HALO (5 MICROSOFT LANG.)	595 389
METAWINDOW PLUS	275 229
TURBOWINDOW/C	95 79
TURBO HALO (FOR TURBO C)	99 79

MODULA-2	
LOGITECH MODULA-2	99 79
COMPILE KIT	249 199
DEVELOPMENT SYSTEM	169 139
TOOLKIT	169 139
REPORTOR	195 169
STONYBROOK MODULA-2	345 299
W/UTILITIES	

OPERATING SYSTEMS	
MICROPORT SYS V/AT	SPECIAL 549 465
SCO XENIX SYSTEM V	1295 99
WENDIN-DOS	SPECIAL 99 79
OTHER MICROPORT, SCO, WENDIN PRODUCTS	CALL CALL

PASCAL COMPILERS	
MARSHAL PASCAL	189 155
MICROSOFT PASCAL	300 189
PASCAL-2	350 319

LIST OURS	
TURBO PASCAL	100 69
TURBO PASCAL DEV. LIB.	395 289
BORLAND ADD-ONS	CALL CALL

OBJECT-ORIENTED LANGUAGES

ACTOR—Powerful new language built around object-oriented programming, with windows being defined as objects. Actor makes it easier to include and control windows in application programs. List: \$495 Special Price: \$409

SMALLTALK/V NEW V. 2.0—New version is a high-performance, production quality object-oriented programming environment. Includes:

- Advanced user interface featuring windows, pop-up menus and optional mouse.
- A set of tools for organizing and browsing the Smalltalk source code.
- An incremental program development capability.
- Bitmap graphics with optional color support.

List: \$99 Special Price: \$79

PFORCE++—This C++ library provides everything necessary to build complete applications. Includes high-level classes for windows, databases, B-trees, fields, menus, rings, lists, communication tasks, time/date stamps, BIOS and DOS access; etc. Complete source code included. List: \$395 Special Price: \$199

ADVANTAGE C++—ADVANTAGE C++ now has MS Windows Support and gives you the speed, support and reliability you need to develop large, complex programs with fewer bugs. Latest version supports MS C 4.0/5.0 and QuickC faster. List: \$495 Special Price: \$469

TURBO PASCAL ADD-ONS	
ASCII TURBO GHOST WRITER	
STARTER	NEW 99 89
COMPLETE	NEW 289 259
AZATAR DOS TOOLKIT	NEW 99 85
DOS/BIOS & MOUSE TOOLS	75 69
FLASH-UP	89 79
METABYTE DATA ACQ. TOOLS	100 89
SCREEN SCULPTOR	125 95
SYSTEM BUILDER	150 129
IMPEX	100 89
REPORT BUILDER	130 115
T-DEBUG PLUS	60 49
TURBO.ASM	99 69
TURBO ASYNCH PLUS	129 99
TURBO GEOMETRY LIBRARY	NEW 100 89
TURBO HALO	99 85
TURBO MAGIC	199 179
TURBO POWER TOOLS PLUS	129 95
TURBO POWER UTILITIES	95 79
TURBO PROFESSIONAL 4.0	99 79
TURBO WINDOW/PASCAL	95 79

SCREEN DISPLAY/WINDOWS	
C-SCAPE	279 265
CURSES W/SOURCE CODE	250 189
GREENLEAF DATA WINDOWS	225 155
W/SOURCE CODE	395 259
HI-SCREEN XL	149 119
JVACC FORMAKER	495 449
JVACC JAM	750 679
MICROSOFT WINDOWS	99 65
MS WINDOWS DEVELOPMENT KIT	500 319
PANEL PLUS	495 395
PANEL/QC OR /TC	129 95
SCREENSTAR W/SOURCE	198 169
VITAMIN C	225 155
VC SCREEN	99 79
VIEW MANAGER	275 199
WINDOWS FOR DATA	295 239
W/SOURCE	NEW 590 CALL

XENIX/UNIX SOFTWARE

MICROPORT + SCO PRODUCTS	CALL CALL
ADVANTAGE C++	695 625
DIRECTORY SHELL (286)	349 315
DIRECTORY SHELL (386)	495 445
FORBASE PLUS	795 649
INFORMIX PRODUCTS	CALL CALL
JVACC FORMAKER	895 809
JVACC JAM	1350 1219
KORN SHELL	145 129
MICROSOFT LANGUAGES	CALL CALL
PANEL PLUS	795 675
RM/COBOL	1250 949
RM/FORTRAN	750 549
WINDOWS FOR DATA	795 CALL

ADDITIONAL PRODUCTS	
ADVANTAGE VCMs	379 329
BASTOC	495 399
CARBON COPY PLUS	195 159
DAN BRICKLIN'S DEMO PROGRAM	75 59
DB2C	299 CALL
FLOW CHARTING II	229 205
MAGIC PC	195 179
MKS TOOLKIT	139 115
MKS RCS	NEW 189 169
NORTON GUIDES	100 65
PC-LINT	139 99
POLYMAKE	149 129
POLYTRON PVCS	CALL CALL
PRE-C	295 159
SOURCE PRINT	95 75
TREE DIAGRAMMER	77 69

Terms and Policies
 • We honor MC, VISA, AMERICAN EXPRESS
 No surcharge on credit card or C.O.D. Prepayment by check. New York State residents add applicable sales tax. Shipping and handling \$3.95 per item, sent UPS ground. Rush service available, prevailing rates.
 • Programmer's Paradise will match any current nationally advertised price for the products listed in this ad.
 • Prices and Policies subject to change without notice.
 • Hours 9AM EST—7PM EST
 • We'll Match any Nationally Advertised Price.
 • Mail Orders include your phone number.
 • Ask for details. Some manufacturers will not allow returns once disk seals are broken.
 • Dealers and Corporate Buyers—Call for special discounts and benefits!

1-800-445-7899

In NY: 914-332-4548

Customer Service:

914-332-0869

International Orders:

914-332-4548

Telex: 510-601-7602

Programmer's

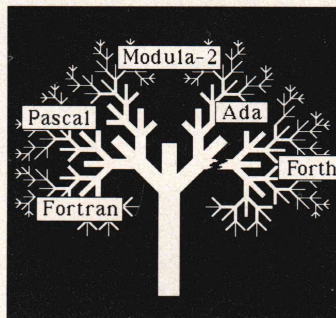
Paradise

A Division of Hudson Technologies, Inc.

42 River Street, Tarrytown, NY 10591



Handling Huge Arrays



Let's face it, when IBM selected the 8088 as the engine for its first-generation PC, memory-segmented architecture officially became a bad idea whose time had come. Three-and-a-half generations of Intel chips later, we're still stuck with it, and there's no relief in sight. Credit for all the remarkable things done on PCs goes not to the architecture but to those with the ingenuity to find ways around the flaw in its addressing scheme: no single hunk of memory in a PC can be bigger than 64K.

Compilers for the PC have dealt poorly with the problem. You can have multiple code segments and in some cases multiple data segments, but the 64K limit still applies to any single extent. Probably someone will write to tell me that the latest release of Googlephonic Snobol or some such has made segmentation transparent, but I'm talking about the mainstream languages from the likes of Microsoft and Borland. They deal with memory segmentation by not dealing with it in any graceful fashion.

The solution is, of course, to keep all objects (code units, global variable sets, arrays, and so on) smaller than 64K. This is usually a workable solution because most objects are inherently smaller than the magic number. In some cases, though, it's simply not possible to remain within this arbitrary limit. Nobody gets hit harder in that regard than scientists and engineers, who routinely work

by Kent Porter

with extremely large numeric arrays.

Many have tried the PC and found segmentation an insuperable barrier, even though their data would, given a decent addressing scheme, fit within the free memory of the average PC. This article is for them and for anyone else faced with the unsa-

very task of attempting to process arrays of several thousand elements.

Memory Segmentation

Memory segmentation works like this: It takes two elements to express a complete address. The first element is a base, or point of reference, and the second is an offset from the base. In the Intel chips, the segment registers (CS, DS, and ES) contain base addresses that remain relatively fixed so that the offsets of variables, entry points, and so on can be reliably measured from them.

A segment address is actually the upper four digits of a five-digit hex number. Therefore, each increment of the segment represents a 16-byte jump in real memory. In the endless quest for quaint terminology, someone has dubbed these 16-byte hunks "paragraphs." A segment always begins at a paragraph boundary, which is a multiple of 16 bytes.

The offset is another four-digit hex value that measures a distance in bytes toward the top of memory, using the segment as the reference point. Four hex digits, of course, can represent any of 64K values. Consequently, any addressing range in the PC is constrained to 64K from a paragraph boundary.

The CS register contains the base address for a code segment. When a program takes a long jump (or far call), the CS register changes to the base paragraph of the new code segment. A far return reinstates the old CS so that execution resumes in the original code space. Thus a program can have multiple code segments, each of up to 64K.

The DS (data segment) register is less flexible. Compilers usually enforce a stable DS throughout program execution so that globals are universally accessible, meaning that the total size of all globals cannot exceed 64K, lest they be beyond the range of the offset.

Auto variables—those allocated as locals within subprograms or passed as parameters—are even more constrained. They're allocated on the stack—another 64K-max structure—so that space is limited to 64K less the stack space already occupied by other stuff.

So what's a body to do with a huge hunk of data? There are a couple of alternatives. One is to build and process array images on disk. That's an ugly way out. You'd better start a big matrix multiplication just before a long weekend. Another option, which I'll explore here, is using the heap.

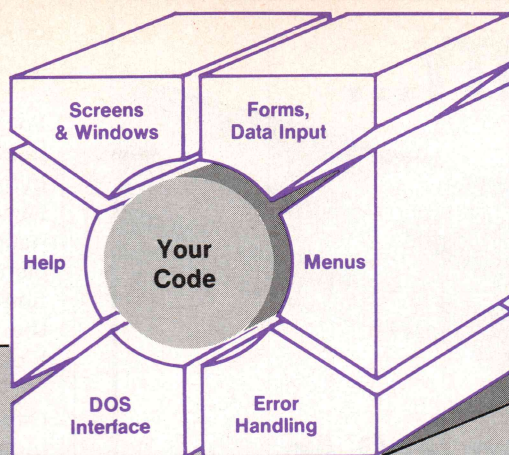
Purists may quibble, but the heap is essentially all the memory that's left after any software currently in memory has staked its claims. Unless specifically set up otherwise, .COM and .EXE programs consider the whole of uncommitted memory as their heap space. Given a moderately sized program and a couple of memory-resident utilities, a 640K machine can usually yield a heap of about 450K to 500K. That's a respectable amount of space: a quarter of a million integers, or upward of 60,000 to 80,000 reals depending on precision. The trick is to use it effectively.

What Can You Put into 64K?

Programming strategies for managing huge arrays inevitably involve some pencil sharpening. Let's first consider how much data can fit inside a 64K node allocated on the heap.

Compilers typically claim a few bytes of memory for record keeping.

30 day moneyback
satisfaction guarantee



C-Worthy Interface Library helps you smoothly pull together all aspects of an excellent Human Interface.

C Programmers: Wrap an Exciting, Bullet-Proof Interface Around Your Code Quickly.

Introducing... C-Worthy® Interface Library

The only human interface package you need. That's what our customers are telling us. One early adopter, Novell, Inc. uses it exclusively in the development of their NetWare® Utilities, which reach over 500,000 users. You see, C-Worthy Interface Library is the only library available to handle every aspect of your program's human interface, all in one package. Now your programs will have a consistent look and feel. You no longer have to integrate pieces of libraries from different manufacturers.

As important as you know users are, you often don't have the time to heavily invest in writing routine code. And that's OK, because with over 400 tight, ready-to-use functions, C-Worthy Interface Library takes care of the tedium and lets you spend your time doing what you enjoy. Concentrate on the heart of your application — features that make it unique, special. Let C-Worthy Interface Library do your:

- Menus
- Error Handling
- DOS Interface
- Context Sensitive Help
- Screens, Windows
- Forms, Data Input (optional)

You control color, size, border, location, etc. And if there's anything you want to change, you can. Source is available to provide you with the flexibility you need. And you can distribute your applications freely, with no royalties.

C-Worthy Interface Library requires hard disk media with 256K RAM. MSDOS 2.0+ and IBM PC, or compatible, TI Professional, NEC APC III, or VICTOR 9000. C-Worthy is a registered trademark of Custom Design Systems, Inc.

Tech Specs

- Compilers: Microsoft 3.0+, Quick, Turbo, Lattice. All models.
- 350+ functions written in C, 75+ in Assembler.
- Menus: Fully support pop-up, Lotus style, MS Windows style (pull-down), pull-up.
- Errors: DOS, program, and user.
- DOS Interface: 62 functions. File handling, dir. and drive management, date & time conversion, wildcards, more.
- Help: System and context sensitive.
- Screens: Screen display, color palettes, save, restore, scroll, more.
- Windows: Exploding, tiled, pop-up, overlapping. Direct video access and virtual. Up to 50 active at any time.
- Keyboard Handling: Regular, function, interrupt, background procedures.
- Editing: String and word wrap text.
- Form Interface Library: 118 functions. Over 15 field types, and user definable field types. 3 levels of data validation: type, multiple field ranges, optional validation procedures. Hide, lock, or secure a field. Optimal field movement.
- Foreign Languages: All text messages in separate files for easy translation.
- Compatible with MS Windows.
- OS/2 special overlay when released.
- Machines: Autodetect for MDA, CGA, EGA, VGA, TI, AT&T, Victor.
- No royalties.

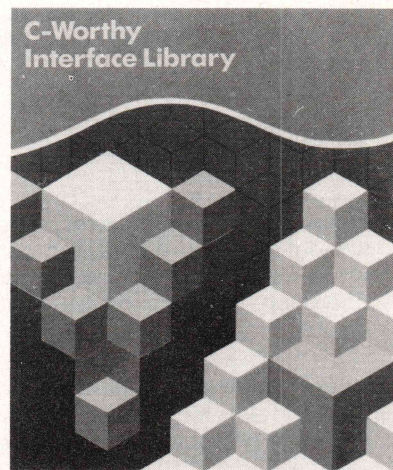
"I heartily recommend this package."

— David A. Schmitt, president, Lattice, Inc.
Over 400 developers in 16 countries already use it.

Thorough Documentation

Indexed alphabetically and by category, the 700+ page Reference Guide includes for each function: an example, description, calling conventions, return values, and related functions. The 250 page User's Guide gets you going with its tutorial and "Getting Started" sections.

"C-Worthy is a comprehensive C library whose time has come. I heartily recommend it as your next purchase." —Computer Language, 8/87



C-Worthy Interface Library:

Object only	\$ 195
Form Interface Library add-on	\$ 100
Object with Forms	\$ 295
Object with Forms & Library Source	\$ 495

Please specify compiler and version when ordering.

To Order Call


(800) 821- 2492

in MA (617) 337-6963

**Solution
Systems™**

541-D Main Street, Suite 410
South Weymouth, MA 02190

ISN'T IT A PITY...



Everything Isn't As Accommodating As

c-treeTM FILE HANDLER

Multi-Key ISAM functions ■ fixed and variable length data records
■ record locking ■ variable length keys and key compression
■ overcomes OS file limits ■ state-of-the-art B+ Trees ■ simplified installation

New version includes c-tree server ■ much faster access to shared files ■ the industry's first portable server runs with Net Bios, UNIX systems, PC-NET compatibles, XENIX systems, Novell (including 2.1 VAP), VAX/VMS ■ server design supports LANS and multi-tasking systems

r-treeTM REPORT GENERATOR

Saves days of coding ■ no printer spacing charts ■ change reports without recoding ■ unlimited control breaks, accumulators and virtual field calculations ■ powerful search, select and sort operations over multiple files ■ automatic file traversal ■ easy report layout

Unparalleled portability ■ our code is running in over 60 system environments ■ DOS, UNIX, OS/2, XENIX, Macintosh, Tower, VAX, CRAY... ■ all popular C compilers on DOS and other OS's

More for your money ■ for one low price, you get complete source code for single and multi-user applications ■ no royalties
■ unlimited tech support ■ free upgrade listings ■ freedom to port our code to all your machines

How to order: Put FairCom's leadership in programmer utilities to work for you. Order c-tree today for \$395 or r-tree for \$295. When purchased together, r-tree is only \$255. For VISA, MasterCard and COD orders, call 314/445-6833. FAX 314/445-9698.



FairCom

4006 West Broadway
Columbia, MO 65203

Complete C Source Code and No Royalties!

The following are trademarks as noted: UNIX/AT&T, XENIX/Microsoft, Inc., NOVELL/Novell, Inc., VAX/DEC, TOWER/NCR Corp., MACINTOSH/Apple Computer, Inc.

CIRCLE NO. 180 ON READER SERVICE CARD

STRUCTURED PROGRAMMING (continued from page 108)

For a large array occupying a heap node, figure the overhead at 16 bytes, which leaves 65,521 for data. Divide that by the size of the array's data type to find out how many elements will fit, truncating any fractional result. Table 1, page 111, lists the sizes of some common data types.

Taking Turbo Pascal reals as an example, $65,521/6 = 10,920$ array elements. That's an approximately square matrix of 104×105 , or one of 26×420 , or any other combination of multiples and submultiples whose product is 10,920 or less. Similarly, a 64K array of IEEE doubles can have 8,190 elements, or 90×91 , 182×45 , 30×273 , and so on.

Assuming that's enough for your matrix, it's easy to make the necessary declarations and allocate the space. You might define the array in Pascal as:

```
TYPE arrayPtr = ^bigArray;  
bigArray = ARRAY [1..90, 1..91]  
OF DOUBLE;
```

and declare a pointer variable as:

```
VAR arr1 : arrayPtr;
```

Allocation is then accomplished with the standard procedure *NEW* (*arr1*); after which you can refer to elements with the notation *arr1*[^] [*row*, *col*].

Pascal/Modula-2 pointer types such as *arrayPtr* are automatically 32-bit *segment:offset* objects. You can therefore allocate several 64K arrays on the heap and, using subscripted pointer notation as above, compare and combine them.

Listing One, page 86, illustrates this with a program named *maddints.pas*. It adds two integer matrices, each 180×180 , and stores the results in a third heap node. An array of 180×180 encompasses 32,400 elements, or 64,800 bytes. There are three such arrays, occupying a total of 194,400 bytes of heap space.

Maddints is somewhat artificial in that it sets both *addend* arrays to the same values, where each element contains a quantity computed as:

$D^{[row, col]} := (row * 10) + column;$

Thus the elements of the first row of both *A* and *B* are 11, 12, 13, ..., of the second 21, 22, 23, ..., and so on, and the elements of the sum array *C* are double those of the sources (22, 24, 26, ...). To make this into a real-world program, revise the *Acquire* procedure to read the data from an external source such as a disk file and add a procedure to output the results to a suitable medium.

Matrices Bigger Than 64K?

Truly huge matrices require a bit more ingenuity to live within the means of 64K segments. In this case, each row of the matrix is a separate array of up to 64K in size, with the subscripted elements representing columns. A matrix of *n* rows consists of *n* such horizontal lists. The glue that holds it together is another array—a vertical list containing *n* pointers to the rows in sequence. Figure 1, this page, illustrates this structure, where *D* points to an array of pointers, each of which points to an individual row.

Conceptually, the matrix elements can be regarded as $D[1..n, 1..z]$, where *n* is the number of rows and *z* the number of columns. Because of pointer following, however, the actual Pascal/Modula-2 notation is more complex. The next couple of paragraphs develop the notation based on a model.

You can define the types for this matrix structure as:

```
TYPE dataObj = WORD; {or whatever type is appropriate}
    colPtr = ^colArray; { pointer to a row }
    colArray = ARRAY [1..maxCols]
OF dataObj; { row }
    colNode = RECORD
        col : colPtr; { row pointer
    node }
    END;
    rowPtr = ^rowArray;
    rowArray = ARRAY [1..maxRows]
OF colNode; { list }
```

The sanity of defining the *colNode* record with only one field will become apparent when you see the actual notation. Also, it seems

contradictory to use the names *colPtr*, *colArray*, and so on in referring to a row, until you realize that you use these objects to identify a specific column within a given row using subscripts.

The variables declared from these type definitions are very simple:

```
VAR A, B, C : rowPtr;
```

That is, the only variables deriving from the types are pointers to the controlling lists for the three matrices *A*, *B*, and *C*. After creating the matrices, you can refer to individual elements by following the pointer structure. For example:

$A^{[25].col^{[287]}}$

refers to the element in row 25,

Type	Bytes
Char, byte	1
Integer, word	2
Long integer	4
Pointer	4
Turbo Pascal real	6
IEEE single	4
IEEE double	8
IEEE extended	10
IEEE comp	8

Table 1: Some common data types and their storage sizes

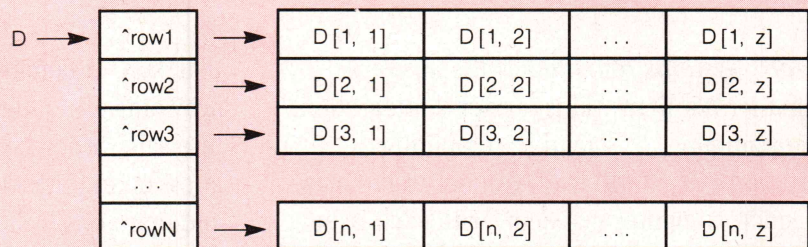


Figure 1: Structure of a huge array

The C Programmer's Assistant

C TOOLSET

Unix-like Utilities for Managing C Source Code

No C programmer should be without an assistant. C ToolSet provides you with the support you need to make C programming easier.

All of the utilities are tailored to the C language but you can modify them to work with other languages as well.

Source code in standard K&R C is included. You are welcome to use it with any compiler (UNIX compatible) and operating system you choose.

The documentation contains descriptions of each program, a listing of program options, and a sample run. On-line help responds to -? on the command line with a list of options

12 Time Savers

DIFF - Compare files line by line; use **CMP** to compare byte by byte.
GREP - Regular expression search.
FCHART - Trace the flow of control between large program modules.
PP - Format C program files so they are easier to read.

CUTIL - General purpose file filter.
CCREF - Cross reference variables.
CBC (curly brace checker) - Check for pairing of curly braces, parens, quotes, and comments.
 Other utilities include **DOCMAKE**, **ASCII**, **NOCOM**, and **PRINT**.

Requires MSDOS and 12K RAM

MONEYBACK GUARANTEE

Try C ToolSet (\$95) for 30 days — if not satisfied get a full refund.

Call (800) 255-4659

In MA (617) 331-0800



The Coder's Source™

541-D Main St., Suite 412, So. Weymouth, MA 02190

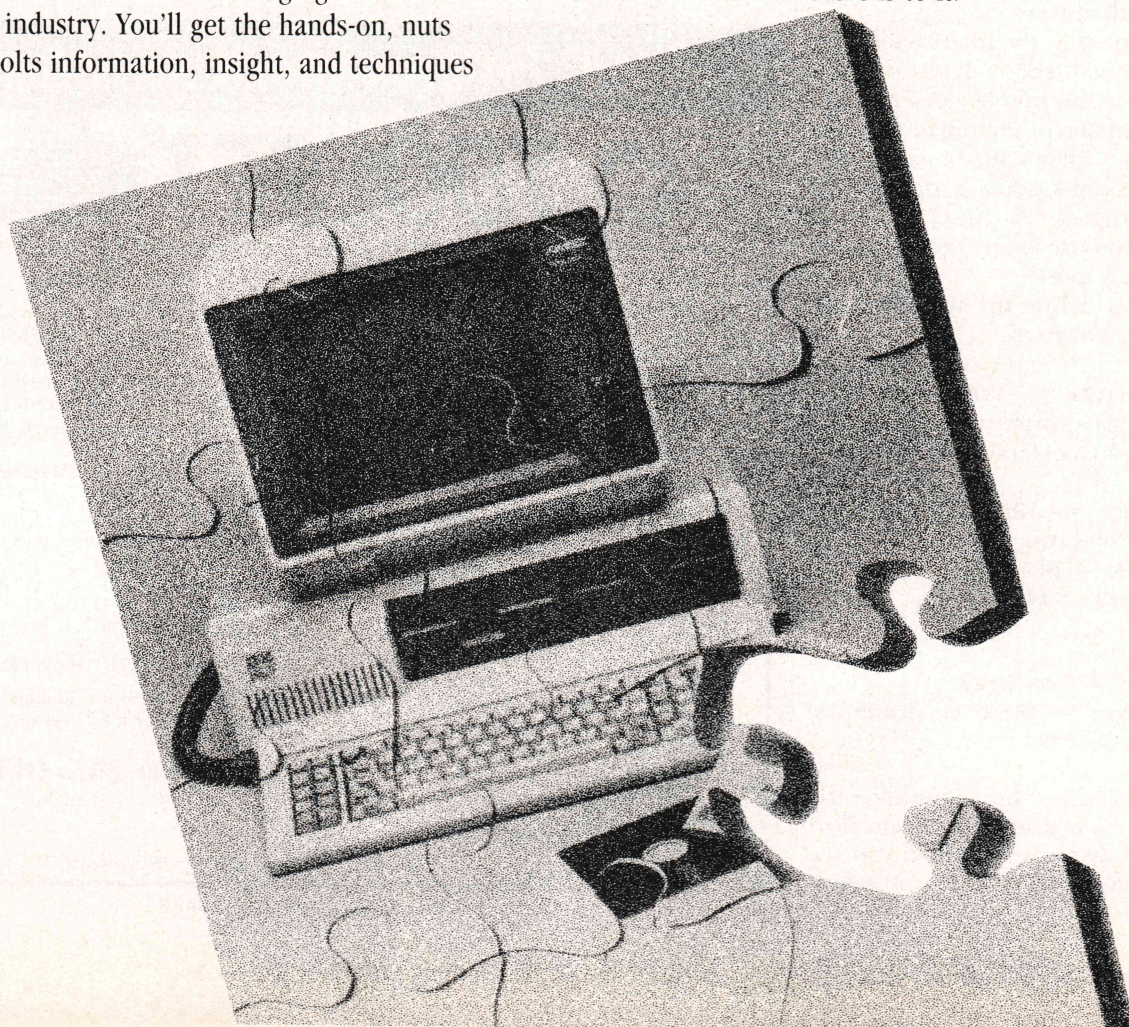
CIRCLE NO. 181 ON READER SERVICE CARD

Why puzzle when you don't have to?



Micro/Systems Journal has the answers. Whether it's networking, systems integration, programming, or scientific computing questions, *M/SJ* will lead you out of the maze of microcomputer mayhem. With each issue you'll find comprehensive coverage of all the technical information that will keep you up-to-date with the ever-changing microcomputer industry. You'll get the hands-on, nuts and bolts information, insight, and techniques

that *M/SJ* is famous for providing . . . in-depth tutorials, reviews, hints, the latest on multitasking, languages and operating systems. So stop your puzzling . . . subscribe right now and the answers will be yours. Simply drop the attached card in the mail—that's all there is to it.



MICRO/ SYSTEMS

JOURNAL
FOR THE
PC SYSTEMS
INTEGRATOR

SUBSCRIBE
NOW AND

SAVE
OVER
47%

OFF THE
NEWSSTAND
PRICE!



Yes! I want to subscribe to
MicroSystems
JOURNAL for the PC Systems Integrator

47%
SAVINGS

and save over 47% off the cover price.

☐ 1 Year (12 issues) \$29.97 ☐ 2 Years (24 issues) \$56.97

Please charge my: ☐ Visa ☐ Master Card ☐ American Express

☐ Payment Enclosed

☐ Bill me later

Card # _____ Exp. date _____

Signature _____

Name _____

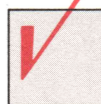
Address _____

City _____ State _____ Zip _____

*Savings based on the full one-year newsstand rate of \$47.70. All foreign countries please add \$7 per year for surface mail; Canada & Mexico add \$17 per year for airmail; other countries add \$28 per year for airmail. All foreign subscriptions must be paid in U.S. funds drawn on a U.S. bank. Please allow 6-8 weeks for delivery.

A PUBLICATION OF M&T PUBLISHING, INC.

211S



Yes! I want to subscribe to
MicroSystems
JOURNAL for the PC Systems Integrator

47%
SAVINGS

and save over 47% off the cover price.

☐ 1 Year (12 issues) \$29.97 ☐ 2 Years (24 issues) \$56.97

Please charge my: ☐ Visa ☐ Master Card ☐ American Express

☐ Payment Enclosed

☐ Bill me later

Card # _____ Exp. date _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

*Savings based on the full one-year newsstand rate of \$47.70. All foreign countries please add \$7 per year for surface mail; Canada & Mexico add \$17 per year for airmail; other countries add \$28 per year for airmail. All foreign subscriptions must be paid in U.S. funds drawn on a U.S. bank. Please allow 6-8 weeks for delivery.

A PUBLICATION OF M&T PUBLISHING, INC.

211S



Yes! I want to subscribe to
MicroSystems
JOURNAL for the PC Systems Integrator

47%
SAVINGS

and save over 47% off the cover price.

☐ 1 Year (12 issues) \$29.97 ☐ 2 Years (24 issues) \$56.97

Please charge my: ☐ Visa ☐ Master Card ☐ American Express

☐ Payment Enclosed

☐ Bill me later

Card # _____ Exp. date _____

Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

*Savings based on the full one-year newsstand rate of \$47.70. All foreign countries please add \$7 per year for surface mail; Canada & Mexico add \$17 per year for airmail; other countries add \$28 per year for airmail. All foreign subscriptions must be paid in U.S. funds drawn on a U.S. bank. Please allow 6-8 weeks for delivery.

A PUBLICATION OF M&T PUBLISHING, INC.

211S



No Postage
Necessary
If Mailed
In The
United States

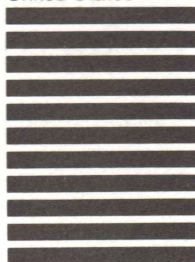
BUSINESS REPLY MAIL

FIRST CLASS PERMIT 790, REDWOOD CITY, CA

Postage Will Be Paid By Addressee

MicroSystems
JOURNAL for the PC Systems Integrator

Box 3713
Escondido, CA 92025-9843



No Postage
Necessary
If Mailed
In The
United States

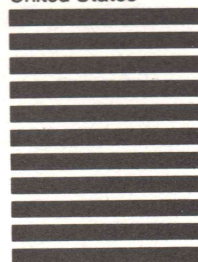
BUSINESS REPLY MAIL

FIRST CLASS PERMIT 790, REDWOOD CITY, CA

Postage Will Be Paid By Addressee

MicroSystems
JOURNAL for the PC Systems Integrator

Box 3713
Escondido, CA 92025-9843



No Postage
Necessary
If Mailed
In The
United States

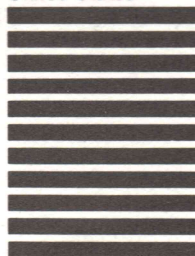
BUSINESS REPLY MAIL

FIRST CLASS PERMIT 790, REDWOOD CITY, CA

Postage Will Be Paid By Addressee

MicroSystems
JOURNAL for the PC Systems Integrator

Box 3713
Escondido, CA 92025-9843



column 287.

An ever-present danger when working with huge arrays is that you'll run out of heap space. Most languages furnish an intrinsic function to inquire about space availability. Logitech Modula-2, for example, has an *AVAILABLE* function that returns a Boolean indicating if the requested number of bytes can be allocated; Turbo Pascal 4.0 has *maxAvail*, which returns the size of the largest contiguous block. Before attempting to allocate space for a huge array, inquire whether your request will be honored. If it won't be, terminate gracefully. This is preferable to simply letting the program crash with some arcane run-time error message.

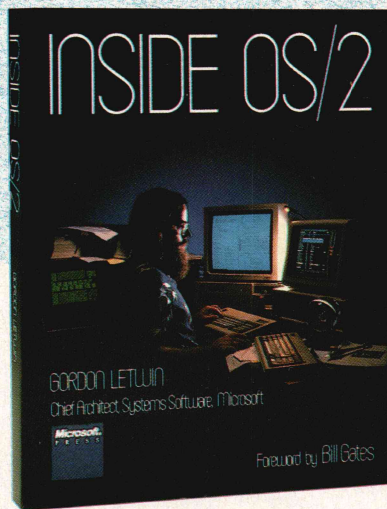
Listing Two, page 86, is a program *hugemats.pas*, written in Turbo Pascal 4.0, that implements this discussion. It's similar to Listing One, except that the matrices are 250 rows by 300 columns, or 75,000 elements apiece. Because the data object is of type *WORD* (an unsigned integer), this comes to 150,000 byte per matrix for data, plus 1,000 byte for the 250 row pointers, for a total of 151,000 byte. There are three such matrices, claiming 453,000 byte of heap space.

That's too much data for the program to run in the Turbo Pascal interactive environment, which provides less than 300K of heap space on a 640K machine. Because of the space checking in the *create* procedure, the program will always terminate with an out-of-memory message if the Turbo Pascal environment is resident. On my machine, the compiled .EXE program running alone has only about 20K of heap space left after the three matrices are created, but that's enough for it to run.

Freeing Up More Space

Huge number crunchers such as those in Listings One and Two are typically bare-bones programs, avoiding slick user interfaces and other memory-gobblers. Remember, every little nicety in a program consumes precious memory and eats into the heap. If you're tight on space, cut

The First Word on OS/2



"During the next 10 years, millions of programmers and users will utilize OS/2... The best way for them to understand the overall philosophy of the system will be to read this book." Bill Gates

INSIDE OS/2. Here—from Microsoft's Chief Architect of Systems Software—is a candid and exciting technical examination of OS/2. In unprecedented detail, Gordon Letwin explores the philosophy, key development issues, programming implications, and future of OS/2. And he provides the first in-depth look at each of OS/2's design elements—how they work alone and their roles in the system. INSIDE OS/2 is a valuable programmer-to-programmer discussion of the graphical user interface, multitasking, protection, encapsulation, inter-process communication, and more. You can't get a more inside view. \$19.95.

Microsoft® Press Quality Computer Books

Available wherever books and software are sold.

Or call in your credit card order. 800-638-3030 (In MD 824-7300). Refer to ad DD38.

CIRCLE NO. 182 ON READER SERVICE CARD

4 TIMES FASTER THAN TODAY'S FASTEST ASSEMBLER!

That's right. 4 times faster.

Clocking in at over 75,000 lines per minute on a 6MHz IBM AT, OPTASM is four times faster than Microsoft's MASM 5.0. 4 times faster — that's 400% more throughput!

But speed is only one part of it. OPTASM is nearly 100% compatible with MASM 5.0 (except 386 support). It is the only single assembler capable of supporting the various incompatibilities between MASM 3, 4 & 5. That makes OPTASM more MASM compatible than any single version of MASM!

Other features? OPTASM generates smaller code without ever generating extra NOP's. It automatically handles jumps out of range, up to 15,000 symbols and most of MASM's phase errors. It also boasts a built in MAKE and simplifies segmentation.

That's why we can make our OPTASM challenge: Test OPTASM head to head against MICROSOFT MASM 5.0. Order both assemblers with their 30-day guarantees. In a lot less than 30 days, you'll see just how dazzling OPTASM's speed really is. You'll realize that we're compatible, easier to use, and deliver many more important features than MASM. So accept our challenge. Try both assemblers. Four times faster and more features, too. We know which one you'll send back.

Write or call us to order or for our detailed brochure.

OPTASM: \$195 Guaranteed returnable within 30 days.

AXSLR

SYSTEMS

CIRCLE NO. 183 ON READER SERVICE CARD



WHAT DO PROGRAMMERS SAY ABOUT OPTASM?

"It (OPTASM) just blows MASM away ... reduces my assemble time for Periscope from 3-plus minutes to less than 45 seconds."

Brett Salter, President,
The Periscope Company

"OPTASM has been absolutely solid. For me, the most useful new product in 1987."

Chris Dunford, Columbia, MD

1622 N. Main Street
Butler, PA 16001
412-282-0864
BBS 412-282-2799
Telex 559215
800-833-3061

out the gimmicks.

The stack is another place you can mine for memory. In Turbo Pascal 4.0, the default stack size is 8,192 byte; it's 8,000 byte in Logitech Modula-2. That's a waste of memory for programs such as those given here, which never have more than about 20 byte on the stack at once. Trim the stack down to a minimal size. The savings go into the heap, where you can use them productively.

Don't forget to free up dynamic space that's no longer needed. For structures such as those given in Listing One, you can loop through *rowArray*, calling *freeMem* in Turbo Pascal (or its equivalent in other dialects) to dispose of allocated nodes, and then free *rowArray* itself.

If your data requirements are still too huge to fit, you'll either have to give up on the PC or else resort to memory-stretching technologies such as EMS (which I'll discuss in a future column). The techniques shown here, however, should satisfy most requirements for working with huge arrays on the PC.

The suggestion for this month's column came from Dr. Don Walters, Professor of Physics at the U.S. Naval Postgraduate School in Monterey, Calif. If you have a programming issue you'd like to see addressed here, drop me a line (no calls, please) in care of DDJ. Or send an MCI message to KPORTER, Mountain View, Calif. No promises, but I'll tackle as many as I can.

Availability

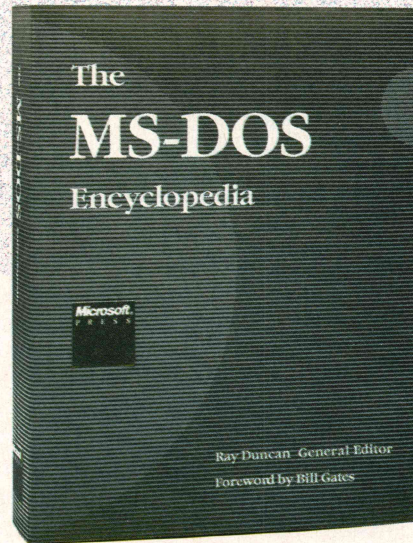
All the source code for articles in this issue is available on a single disk. To order, send \$14.95 to *Dr. Dobb's Journal*, 501 Galveston Dr., Redwood City, CA 94063, or call (415) 366-3600, ext. 216. Please specify the issue number and format (MS-DOS, Macintosh, Kaypro).

DDJ

(Listings begin on page 86.)

Vote for your favorite feature/article.
Circle Reader Service No. 5.

The Ultimate MS-DOS® Resource



MS-DOS — the starting place for the development of over 20,000 software applications. THE MS-DOS ENCYCLOPEDIA — the starting place for data, advice, and insight that will make your MS-DOS programs reliable, robust, and efficient. THE MS-DOS ENCYCLOPEDIA is a programmer's dream. 1600 pages packed with version-specific data. Annotations of more than 100 system function calls, 90 user commands, and a host of utilities. Hundreds of hands-on examples and thousands of lines of code. Plus articles on debugging, installable device drivers, TSRs, Windows, memory management, the future of MS-DOS, and much more. Researched and written by a team of MS-DOS experts — many involved in the creation and development of MS-DOS. Covers MS-DOS through version 3.2, with a special section on version 3.3. **\$134.95.**

Microsoft® Press

Quality Computer Books

Available wherever books and software are sold.

Or call in your credit card order. 800-638-3030 (In MD 824-7300). Refer to ad DD38.

CIRCLE NO. 184 ON READER SERVICE CARD

EXAMINING ROOM

Turbo Pascal, Version 4.0

Target:

IBM PS/2, PC AT, and true compatibles

Requires:

DOS 2.0 or later; 384K for the integrated environment or 256K for the command-line environment

Price:

\$99.95

Vendor:

Borland International, 4585 Scotts Valley Dr., Scotts Valley, CA 95066;
(408) 438-8400

With Version 4.0, Borland has turned its highly successful Turbo Pascal into a language package well suited to serious software development. Until now, Turbo Pascal has been a structured language suited for small, individually written projects. Previous versions have had certain limitations that barred them from consideration for major software projects. With Version 4.0, Borland has given us a Pascal that is still every bit as useful to its original market and that is also a worthy compiler for commercial and academic programmers.

Turbo Pascal 4.0 offers several new enhancements both to the Pascal language and to its programming environment. One example, the graphics unit, furnishes the most comprehensive set of screen-handling routines that I've seen with any general-purpose programming language. I'll discuss it presently, but first I'll look at some of the particulars.

Turbo Pascal 4.0 corrects the following oft-heard complaints regarding earlier versions:

- Each code and data segment limited to 64K: Version 4.0 code can be of any size up to the total of avail-

able memory. The data segment is still limited to 64K of globals, as it is with most other compilers for the PC, but as with earlier releases, you can overcome the 64K data limit by using dynamic allocation.

- No modular compilation: Version 4.0 encourages programmers to use separately compiled units. Programs thus become collections of precompiled modules (.TPU files) glued together by a main program.

- No object libraries: Version 4.0 furnishes utilities for managing compiled object libraries.

- Produces only .COM files: Version 4.0 produces .EXE files, and no separate link step is required. For equivalent programs compiled with other releases, the .EXE file is considerably smaller.

- Inserts unnecessary code into executable files: Version 4.0 optimizes the code, removing unnecessary instructions and unused routines.

- Not compatible with standard Pascal: Version 4.0 is almost compatible with the ANSI/IEEE770X3.97-1983 standard. The well-documented list of exceptions runs at less than two pages, many of them overriding counterproductive requirements of the standard. For example, the standard

forces a program to abnormally end (abend) if there is no clause for the selector's current value in a *CASE* statement; Turbo Pascal 4.0 just falls through the *CASE* unless there's an *ELSE* clause.

Three levels of compilation are available. At the lowest level is a simple compile of the current unit. Next higher is *Make*, which recompiles the current program and any units that have been changed since the last compile. At the highest level is *Build*, which unconditionally compiles all the parts of an application.

The complete user interface is awakened by the *TURBO* command, which starts *TURBO.EXE* running. For hairy-chested traditionalists, a command-line compiler (*TPC.EXE*) is also available, replete with Unix-like switches. I can't imagine why anyone would prefer *TPC*; it doesn't furnish nearly as much functionality (although the language is exactly equivalent), and command-line compilers aren't much fun.

Whichever, compile speed is blazingly fast. Borland claims 27,000 lines per minute on an 8-MHz AT, which is the machine I have. Although I'm unwilling to write 27,000 lines of code simply to prove the company right or wrong, I'll say this: a 600-line program compiles fully before my finger leaves the command key. Even on an old 4.77-MHz XT, Turbo Pascal 4.0 compiles equivalent code as or more rapidly than Turbo C on the AT. And Turbo C is no slouch.

Linking

Both the integrated and command-line compilers take the somewhat unusual approach of incorporating the linker. This substantially decreases the overall cycle time from .PAS to resulting .EXE. The built-in linker brings in .OBJ files in standard Intel relocatable format; it iden-

Ron Copeland, associate editor, is the coordinator for this review section. He welcomes your feedback on products worth reviewing.

If you ever wanted to take a crack at assembly language, now's the time.

You probably already know that assembly language subroutines are the smartest way to get the fastest programs.

But if the complexities of working in assembler made you think twice, here's some good news. We've made Microsoft® Macro Assembler Version 5.0 a lot easier to use.

We eased the learning process by giving you the best support around. We completely revised our documentation. The new Mixed Language Programming Guide gives you step by step instructions for linking your assembly code with Microsoft QuickBASIC, C, FORTRAN, Pascal and other languages. And you get a comprehensive reference manual with listings of the instruction set and examples of each instruction. We didn't stop there, though. You also get an on-disk collection of templates and examples.

We've also dramatically simplified the high-level language interface. In just a few

simple steps, you can be calling Macro Assembler subroutines from programs written in your favorite language.

Now that you're writing the fastest programs, Microsoft is giving you the fastest way to debug them. For the first time, we've added our CodeView® debugger to Macro Assembler.

With source code and comments on your screen, Microsoft CodeView makes debugging programs containing assembly language subroutines a snap.

And you'll be glad to know that you don't sacrifice any speed for all the ease of use.

We took the fastest Macro Assembler on the market and made it even faster.

So what are you waiting for? Get your hands on Microsoft Macro Assembler and see what it's like to break your personal speed limit.

Microsoft®

For more information or for the name of your nearest Microsoft dealer, call (800) 426-9400. In Washington State and Alaska, (206) 882-8088. In Canada, call (416) 673-7638.

Microsoft, the Microsoft logo and CodeView are registered trademarks of Microsoft Corporation.

CIRCLE NO. 185 ON READER SERVICE CARD

EVEN MORE POWER AND FLEXIBILITY

BRIEF 2.0

Users and industry press alike have unanimously proclaimed BRIEF as the best program editor available today. Now, the best gets better, with the release of BRIEF 2.0.

Straight from the box, BRIEF offers an exceptional range of features. Many users find that BRIEF is the only editor they'll ever need, with features like real, multi-level Undo, flexible windowing and unlimited file size. But BRIEF has tremendous hidden power in its exclusive macro language. With it, you can turn BRIEF

into your own custom editor containing the commands and features you desire. It's fast and easy.

Jerry Pournelle, columnist for BYTE magazine summed it all up by saying BRIEF is, "Recommended. If you need a general purpose PC programming editor, look no further." His point of view has been affirmed by rave reviews in C JOURNAL, COMPUTER LANGUAGE, DR. DOBB'S JOURNAL, DATA BASED ADVISOR, INFO WORLD AND PC MAGAZINE.

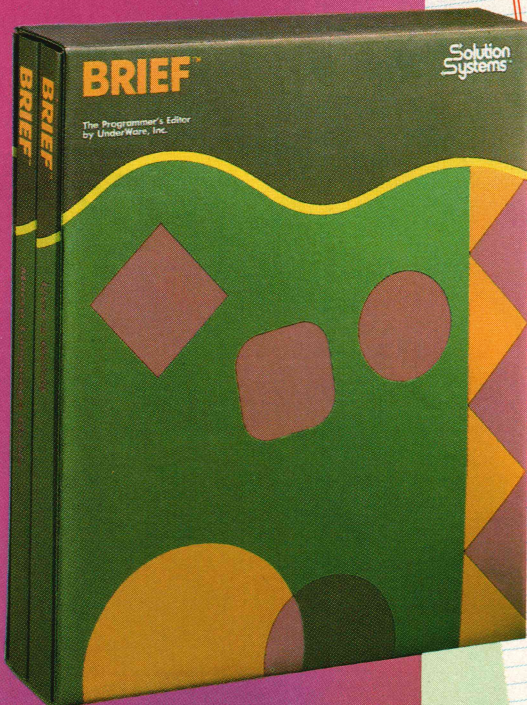
One user stated "BRIEF is one of the few pieces of software that I would dare call a masterpiece." Order BRIEF now and find out why. BRIEF 2.0 is just \$195. If you already own BRIEF, call for upgrade information.

**TO ORDER CALL: 1-800-821-2492
(in MA call 617-337-6963)**

As always, BRIEF comes with a 30 day money-back satisfaction guarantee.

**Solution
Systems™**

541 Main Street
Suite 410D
So. Weymouth, MA 02190
(617) 337-6963



Look at these BRIEF 2.0 enhancements!

Main Features:

- All new documentation with tutorials on basic editing, regular expressions **and** the BRIEF Macro Language.
- Setup program for easy installation and configuration. (Requires no knowledge of the macro language)
- Increased speed for sophisticated operations like Undo and Regular Expression Search.
- Expanded regular expressions, with matching over line boundaries.
- More block types, with marking by character, line or column.
- Command line editing (move cursor, add and delete characters, specify command parameters).
- Support for more programming languages.
- Optional borderless windows.
- Enhanced large display support, including wider displays.
- Reconfigurable indenting for C files (supports most indenting styles).

Plus the basic
features that made
BRIEF SO popular!

Basic Features:

- Full multi-level Undo
- Windows
- Edit many files at once
- File size limited only by disk space
- Automatic language sensitive indentation

Requires an IBM PC or compatible with
at least 192K RAM.
BRIEF is a trademark of UnderWare, Inc.
Solution Systems is a trademark of Solution Systems.

CIRCLE NO. 186 ON READER SERVICE CARD

tifies them via the `$L` switch, which names the `.OBJ` file. This allows you to link with modules written in assembly language or in C using the Pascal calling conventions.

Oddly, the compiler doesn't convert Pascal source files into `.OBJ` format. Source files beginning with the `UNIT` keyword become `.TPU`-compiled files, and those starting with the `PROGRAM` keyword become `.EXEs`. The linker automatically merges any `.TPU` units into the final product, identifying them from

eral different ways. One way marks up the original source file, pointing out obsolete Version 3.0 conventions and explaining what needs to be done. For experienced 3.0 programmers, this is a good way to find out where the mosquitoes are hiding, waiting to bite. Another method merges and unifies the source files for overlays and their parent in order to create a monolithic (multiple code segment) application. There are other options as well. `UPGRADE` is a forward-compatibility

tool.

For backward compatibility, Version 4.0 offers two units called `Turbo3` and `Graph3`. These allow a Version 4.0 program to employ 3.0 features such as turtle graphics, which Version 4.0 doesn't support.

The manual has a 20-page chapter devoted to converting from Version 3.0 to 4.0 in addition to a 12-page appendix detailing the differences between the two. That ought to give you some idea of the extent to which Turbo Pascal has been ex-

For backward compatibility, Version 4.0 offers two units called Turbo3 and Graph3

the `USES` statement.

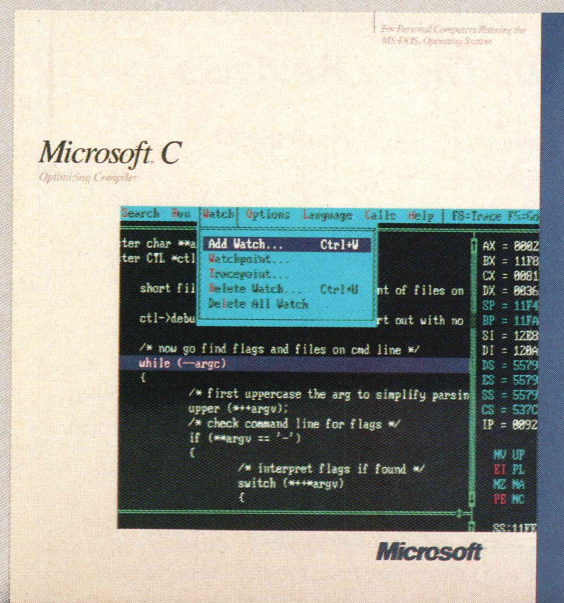
Although this unconventional approach is valid most of the time, it makes an assumption that might not be universally true: that you want only to link foreign-language modules into Pascal programs and not the other way around. The absence of `.OBJ` output files and of any option to create them makes it impossible to export compiled Turbo Pascal modules to other languages. In effect, the built-in linker creates a one-way street leading from other languages to Turbo Pascal. You could wish it were a two-way street.

Compatibility with Version 3.0

Speaking of two-way streets, Version 4.0 does better with regard to Turbo Pascal 3.0 compatibility. Both upward and downward options are available.

A utility program called `UPGRADE`, distributed with Version 4.0, processes Version 3.0 source files in sev-

C5.0
has three features
professional
programmers can't
live without.



Simply the BEST C and Pascal on AT, 386, Sun, Apollo, RT, VAX, 370

"The most rock-solid C compiler in the industry. Superb technical support and portability. Superior code generated."

Gordon Eubanks, Symantec — Q&A (386).

"It simply works, with no trouble, no chasing strange bugs, and excellent warning and error messages ... a professional product."

Robert Lerche, Bay Partners.

"For large-scale software development, the highest quality C compiler available on the market today. Pragmas are great. Quality of support is exceptional." **Randy Neilsen, Ansa—Paradox (DOS, OS/2).**

"15% smaller and 15% faster than Lattice C."

Robert Wenig, Autodesk.

"Our software is running anywhere from 30 to 50% faster than when compiled under Lattice." **David Marcus, Micronetics.**

"We switched from Lattice due to a 10% reduction in code size. The compiler is very stable." **Lee Lorenzen, Ventura Software — Ventura Publisher, marketed by Xerox Corp.**

"Best quality emitted code by any compiler I've encountered. Often amazing." **Bill Ferguson, Fox Software — FoxBase (386).**

"Messages sometimes pointed out type mismatches, incorrect-length argument lists, and uninitialized variables that had been undetected for years [in 4.x bsd]." **Larry Breed, IBM ACIS [RT PC].**

"Diagnostics turned up bugs missed by other compilers. Rapid bug fixes by technical support, someone who knew what he was talking about. 80386 code is well optimized."

Tim Addison, Logistics Data Systems.

"386 protected mode support is fantastic, especially the access to large amounts of memory. It's mainframe compute power on a PC."

Dan Eggleston, Viewlogic.

"The preprocessor supplied with Professional Pascal is quite useful. The code quality and control over segmentation and memory models are superior to MS Pascal." **Bob Wallace, QuickSoft.**

Check Out These Reviews

• High C™:

<i>Computer Language</i>	February 1986, '87	
<i>Dr. Dobbs's Journal</i>	August 1986	
<i>PC Magazine</i>	Jan. 27, 1987	(80386 version)
<i>Dr. Dobbs's Journal</i>	July 1987	(80386 version)
<i>BYTE Magazine</i>	November 1987	(80386 version)

• Professional Pascal™:

<i>PC Magazine</i>	Dec. 29, 1985
<i>Computer Language</i>	May 1986
<i>PC Tech Journal</i>	July 1986
<i>Journal of Pascal, Ada, & Modula-2</i>	Nov.-Dec. 1986
<i>BYTE Magazine</i>	Dec. '86, June '87 (80386 version)

Why MetaWare compilers

- They are specifically designed for serious software developers.
- They are reliable and robust: they don't break at every turn.
- Their generated code is the best, or near best, on each architecture.
- Their superior diagnostic messages help you produce better products more quickly.
- Your source can be ported with ease to the most popular systems.
- You can link mixed-language modules from our compilers, others.
- You can benefit from high-level, personal technical support.
- You can take advantage of the latest ANSI C extensions, and/or extensive Pascal extensions. **High C** has been tracking the ANSI Standard for two years; **Professional Pascal** will soon have a VS Pascal compatibility switch and several Apollo Pascal ext'ns.
- You can take advantage of the latest 387 and Weitek 1167 support — we have the only compilers with Weitek real mode support.



Power Tools for Power Users

Ashton-Tate: dBase III Plus, MultiMate; Autodesk: AUTOCAD, AUTOSKETCH (8087, '387, Weitek); Boeing Computer Services (Sun); CASE Technology (Sun); CAD/CAM giant Daisy ('86, '386, VAX); Deloitte Haskins & Sells; Digital Research: FlexOS; GE; IBM: 4.3/RT, 4680 OS; Lifetree Software (Pascal); Volkswriter Deluxe; GEM-Write; Lugaru: Epsilon; NYU: Ada-Ed cmplr; Semantec: Q&A; Sky Computers; ... (Product names are trademarks of the companies indicated.)

Industrial-Strength

MetaWare C and Pascal compilers are designed for professional software developers. These tools are loaded with options to control them for special purposes. You can adjust the space-time trade-off in code quality. You can adjust external naming conventions to agree with linkers and operating systems. You can specify segment names for segmented architectures, and to help place code or data in particular places for embedded applications. You can select from five memory models for the 8086 family. And on and on.

A Partial List of Optimizations

Common subexpression and dead-code elimination, retention and re-use of register contents, jump-instruction size minimization, tail merging (cross jumping), constant folding, short-circuit evaluation of Boolean expressions, strength reductions, fast procedure calls, automatic mapping of variables to registers (where advantageous), ...

"Platform" — Code Quality

Sun, Apollo, SGI — 18%, 3%, 26% > resident compiler (Dhrystone).
 PC: DOS, OS/2 — 3-10% > Microsoft C; 30% > MS Pascal, Lattice C.
 386 32-bit DOS — no competitors, since November, 1986.
 286, 386 UNIX — 66% better than pcc (Dhrystone, 386).
 VAX VMS — ≈ DEC's excellent C and Pascal; better features.
 VAX Ultrix — 19% > pcc (Dhrystone); much > Berkeley Pascal.
 RT PC/4.3bsd — 89% > the original port of pcc (Dhrystone).
 370 CMS, UNIX — much better than any C, and VS Pascal.
 AMD 29000 — >40,000 Dhrystones! Available in Q2, cross.

(408) 429-6382, telex 493-0879.

Since 1979.



903 Pacific Avenue, Suite 201 • Santa Cruz, CA 95060-4429

The Clear Choice for Large Programming Projects — PC Tech J.

© 1987 MetaWare Incorporated. MetaWare, High C, Professional Pascal, and DOS Helper are trademarks of MetaWare Incorporated. Others and their owners are duly respected.

panded and enhanced in Version 4.0.

Memory Management

While allowing executable code of any size, Version 4.0 is, of course, subject to the 64K segmentation imposed by the Intel processors. It gets around this by limiting the code in any compile unit to 64K and starting a new code segment for each module during linkage. Thus, intrasegment calls (to local subroutines) are near calls employing 16-

ables are always 16-bit offsets relative to *DS*. As in most high-level languages (and identically to Version 3.0), local variables for subprograms are allocated on the stack relative to the *BP* register and so do not count against the 64K limit for globals. Pointers to dynamic objects allocated on the heap are 32-bit segment:offset variables, allowing them to be passed freely among various code segments.

The default stack for a 4.0 program is 8K, and the heap is all of

uncommitted memory above the stack. This eliminates the need to be concerned (as you were in Version 3.0) about heap/stack collisions, but of course it's possible for a highly recursive program to drive the stack down into the data segment. An option lets you set the stack size to something besides 8K and also to claim a finite area for the heap. The latter is important in multitasking and TSRs, neither of which can assume that they own all of memory.

**An option
lets you set
the stack size
to something
besides 8K**

bit offset pointers and those to external routines and to subprograms named in the interface portion of units are implicitly far (32-bit segment:offset) calls. Programmers have no control over these compiler-generated calling conventions.

You can, however, force a routine—or an entire compile unit—into far mode with the *\$F* switch, in which *\$F+* turns on far calls/returns and *\$F-* reverts to default (compiler-controlled) calling conventions according to the rules above. An example of needing a far routine is when furnishing a mouse event handler: the handler is called as a far procedure from the device driver in low memory and thus must return with *RETF*. If the procedure heading is surrounded by the *\$F* switches, its entire body is forced to far.

The *DS* register holds constant throughout the code, no matter how many code segments there are. Consequently, global data is limited to 64K, and references to global vari-

Speed.

Fast Execution Speed.

	Microsoft® C 4.0	Microsoft C 5.0
Sieve (25 iterations)	5.7	3.3
Loop	11.0	0.0*
Float	19.9	0.1
Dhrystone	22.8	19.1
Pointer	14.2	7.4

- New optimizations generate the fastest code:
 - Inline code generation. **NEW!**
 - Loop optimizations: **NEW!**
 - Loop invariant expression removal. **NEW!**
 - Automatic register allocation of variables. **NEW!**
 - Elimination of common sub expressions.
 - Improved constant folding and value propagation.
- Fine tune your programs for even greater speed:
 - Coding techniques for writing the fastest possible programs are included in the documentation. **NEW!**
 - Segment Allocation Control:
 - Group functions into the same segment to get faster NEAR calls. **NEW!**
 - Specify which segments receive variables to yield faster NEAR references. **NEW!**
 - Uses register variable declarations.
 - Mix memory models using NEAR, FAR & HUGE pointers.

*Time is negligible.

Microsoft C 5.0

Optimizing Compiler

The Advanced
Programmer's Editor
That Doesn't Waste Your Time

For DOS, Microport
UNIX, SCO Xenix or

OS/2
Protected Mode

EPSILON

- Fast, EMACS-style commands—completely reconfigurable
- Run other programs without stopping Epsilon—concurrently!
- C Language support—fix errors while your compiler runs
- Powerful extension language
- Multiple windows, files
- Unlimited file size, line length
- 30 day money-back guarantee
- Great on-line help system
- Regular Expression search
- Supports large displays
- Not copy protected

Only \$195

LUGARU
Software Ltd.

5843 Forbes Avenue
Pittsburgh, PA 15217

Call
(412) 421-5911

for IBM PC/XT/AT's or compatibles

CIRCLE NO. 188 ON READER SERVICE CARD

EXAMINING ROOM
(continued from page 121)

An important addition to the Version 4.0 arsenal is a special interrupt-class procedure, which allows you to embed custom interrupt-service routines into your applications. These can be chaining routines ("hooks") that intervene in standard system interrupts, replacements for such things as the critical error handler, or new software- and hardware-driven routines activated via uncommitted vectors. An interrupt procedure looks much like a normal Pascal procedure, except that the *INTERRUPT* keyword follows the heading. Also, the heading can name the CPU registers you want preserved on the stack (subject to sequencing rules), and you can then read and write them as local variables.

A particularly nice feature is that the entry processing points the *DS* register to your program's data segment, furnishing direct access to all the program's globals. Because an interrupt procedure returns via an *IRET* after reloading the parametric registers from the stack, you cannot call it like a normal subroutine from within your program. That's as it should be. This feature alone makes Turbo Pascal, Version 4.0, a worthwhile compiler for serious developers.

Graphics

Another enhancement that scores high on the programmer's Richter scale is a new graphics unit. Encompassing some 60 functions and procedures, it provides a comprehensive set of primitives for controlling the display. These include optional automatic detection of the system's video capabilities and selection of the "best possible" mode among CGA, EGA in its many incarnations, VGA, Hercules, AT&T, and PC3270. You can also inquire about the display options available and force a particular mode selection.

Once in a graphics mode, the graph unit provides various line styles and widths as well as drawing and filling routines for common visual objects (bars, circles, polygons, rectangles, and so on). You can partition off viewports, set clipping

New! Hire a Pro for Your New Turbo 4.0

Turn on the power of Turbo PROFESSIONAL 4.0, a library of more than 300 state-of-the-art routines optimized for Turbo Pascal 4.0. You'll have professional quality programs finished faster and easier.

Turbo PROFESSIONAL 4.0 includes complete source code, comprehensive documentation and demo programs that are powerful and useful. The routines include:

- Pop-up resident routines
- BCD arithmetic
- Virtual windows and menus
- EMS and extended memory access
- Long strings, large arrays, macros, and much more.

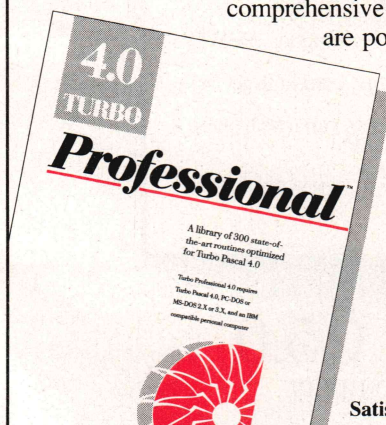
Turbo PROFESSIONAL is only \$99.

Call toll free for credit card orders.

1-800-538-8157 extension 830

1-800-672-3470 extension 830 in CA

Satisfaction Guaranteed or your money back within 30 days.



Turbo Pascal 4.0 is required. Registered owners of Turbo Professional by Sunny Hill Software may upgrade for \$30. Include your serial number.

For other information call 408-438-8608, 9 AM to 5 PM PST. Shipping & taxes prepaid for US and Canadian customers, others please add \$6 per item.

TURBO
Power

TurboPower Software 3109 Scotts Valley Dr., Suite 122 Scotts Valley, CA 95066

CIRCLE NO. 189 ON READER SERVICE CARD

boundaries, inquire about the screen's aspect ratio, monitor error status, and perform a host of other useful graphics operations. The graph unit supports *bitblt* operations for high-speed image transfers, including *XOR*, *OR*, *NOT*, and *AND*. This is a substantial improvement over Version 3.0's *PutPic/GetPic* operations, providing for much more sophisticated animation effects.

Also furnished with the graph unit are four stroked fonts: triplex, small, sans serif, and gothic. Unlike conventional bit-mapped character fonts, stroked fonts allow for justified and variable text sizes up to 10× that can originate at any pixel (rather than character cell) position and can be oriented either horizontally or vertically.

By default—in both text and graphics modes—Version 4.0 writes directly to screen memory rather than going through conventional DOS/ROM BIOS calls. This makes for extremely fast output. You can override the default, and a *CheckSnow* procedure is thoughtfully provided to prevent the unsightly “snow” occurring on older CGAs during direct screen writes. Nevertheless, the line-drawing routine is not particularly fast.

The 4.0 graph unit is impressive, providing the kind of display control and intrinsic graphics calls that make visual programming fun and easy. Unfortunately, it doesn't conform to any graphics interface standard (GKS, PHIGS, or whatever) and thus lacks high-level routines for axis rotation, scaling, and other coordinate transformations. Still, just about everything you need for graphics power is there.

Data Types

Version 4.0 offers several new data types. There are still six basic types—string, char, Boolean, pointer, integer, and real—but the latter two have more choices.

The *integer* types and their characteristics are shown in Table 1, this page. The fundamental floating-point type in Version 4.0 is the same 6 found in earlier versions. The old Turbo-87 became a has-been with

Type	Range	Size
byte	0..255	1
shortint	-128..127	1
integer	-32768..32767	2
word	0..65535	2
longint	-2147483648..2147493647	4

Table 1: Integer types and their characteristics

Speed.

Fast Compilation. Fast Prototyping.

Microsoft C Version 5.0 includes QuickC™, which lets you edit, compile, debug, and execute in an integrated environment. It's ideal for prototyping.

- In-memory compilation at 10,000 lines/minute. **NEW!**
- Built-in editor with parentheses, bracket and brace matching.
- Use the integrated debugger to animate through your program, add watch variables and set dynamic breakpoints. **NEW!**
- MAKE file is automatically generated for you. Simply indicate the modules you want to use, then MAKE recompiles and links only those modules that have changed. **NEW!**
- Full C 5.0 compatibility:
 - Completely source and object code compatible.
 - Emits CodeView®-supported executables.
 - Identical compile/link command line switches.

Microsoft C 5.0 Optimizing Compiler

It's good
for your system!

Vitamin C

"If you need source code, make sure
your wallet is wide open or get
VITAMIN C.

Picking the best value package is hard...
If you're a source code fanatic like me,
VITAMIN C is preferable."

- Computer Language, June, 1987

Fast, flexible, versatile, reliable. Vitamin C delivers the vital combination software professionals demand to produce superior applications in dramatically less time. Highly efficient, professionally crafted C code provides lightning fast displays required by today's window intensive programs.

High level functions provide maximum productivity and require little supporting code. Extended versions of these routines add flexible control over specific details when necessary. Plus, Vitamin C's versatile, open ended design is full of hooks so you can intercept and plug-in special handlers to customize or add features to most routines.

VCScreen, our screen painter / code generator speeds your development even more! Simply draw your input forms using our interactive design editor and generate perfect C source code ready to compile and link with the Vitamin C library.

Vitamin C \$225
Includes all source code FREE! For IBM
PC, XT, AT, PS/2 and true compatibles.
Specify compiler when ordering.

VCScreen \$99.95
For IBM PC, XT, AT, PS/2 and true
compatibles. Requires Vitamin C.

We ship UPS. Please include \$3 for
ground, \$6 for 2-day air, \$20 for
overnight, or \$30 if outside the U.S.
Texas residents add 7 1/4 % sales tax. All
funds MUST be in U.S. dollars drawn on a
U.S. bank. Visa & MasterCard accepted.

ORDER NOW!
(214)416-6447

creative
PROGRAMMING

Box 112097 • Carrollton, Texas 75011

- ☒ Professional C function library
- ☒ 30 day money back guarantee
- ☒ Multiple bullet proof windows
- ☒ Easy full screen data entry
- ☒ Unlimited data validation
- ☒ Context sensitive help manager
- ☒ Menus like Lotus and Mac
- ☒ Programmable keyboard handler
- ☒ Text editor routines
- ☒ No royalties or runtime fees
- ☒ Library source included FREE
- ☒ Free technical support
- ☒ Free BBS at (214)418-0059
- ☒ Supports all major compilers
including Microsoft 5.0
- ☒ VCScreen code generator too!
- ☒ UNIX version available,
call for details

Windows • Data Entry • Menus • Help • Text Editing
Plus... All Source Code FREE!

Version 4.0, which offers four additional *real* types compatible with IEEE Standard 754 and usable only on an 80x87 math coprocessor; no IEEE emulation package is available. What's more, the compiler won't let you declare variables of these new types unless you specify the \$N switch, explicitly stating that your target system has a coprocessor. Table 2, this page, gives the available real types.

Pointers are ordinarily bound to a type, but they can be passed as untyped parameters to a function or procedure. One unfortunate exception to the ANSI standard is that Version 4.0 doesn't support the passing of procedural and functional parameters; that is, procedure *B* can't accept a pointer to function *A*, then pass control to function *A* using the pointer. You couldn't do this in earlier Turbo Pascal versions either, so nothing's changed in that regard, but because the standard mandates it and because it's one of those things that makes C such a powerful systems programming language, Version 4.0 should have supported it.

Here's the Wrap

From the beginning, Turbo Pascal has been controversial on two grounds: its limitations and its disregard of formal standards. With Version 4.0, Borland has removed the limitations. As for standards, you could take the position that with more than 600,000 legal copies available—probably more than all other Pascal compilers combined—Turbo Pascal is the standard by sheer weight of numbers. Frankly, that's what I expected Borland to do inasmuch as success breeds arrogance, of which Philippe Kahn has never been found wanting. Instead, though, Version 4.0 has moved much closer to the formal standard, deviating—though differently in particulars—to approximately the same extent as Microsoft, VAX, and other "purer" Pascal dialects. And that's good for everybody.

I wish Turbo Pascal 4.0 had a debugger and that it exported compiled modules for linking with other languages. But what the heck, it's

Type	Range	Digits	Size
real	10**−38. .10**38	11	6
single	10**−38. .10**38	7	4
double	10**−38. .10**38	15	8
extended	10**−4391. .10**4391	19	10
comp	2**63. .2**63−1	8	??
			18.9 digits of precision

Table 2: Real types and their characteristics

And speed.

Fast Debugging.

Microsoft C Version 5.0 includes Microsoft CodeView, our source-level windowing debugger that lets you debug more quickly and thoroughly than ever before.

- Debug larger programs:
 - Debug through overlays created by the Microsoft overlay linker. **NEW!**
 - Expanded Memory Specification (EMS) support. **NEW!**
- Fast debugging through precise control of your program execution:
 - Access source level and symbolic debug information from your Microsoft C, FORTRAN, and Macro Assembler programs. **NEW!**
 - View your source code and assembly simultaneously.
 - Watch the value of variables change as you execute.
 - Set conditional breakpoints.
 - Animate or single step through your program.
- CodeView brings you as close as you've ever been to your hardware:
 - Swap between your code and output screens.
 - Watch your registers and flags change as your program executes.

All benchmarks run on an IBM® Personal System/2.*

Microsoft

For your free C 5.0 information packet, call:

(800) 426-9400.

In Washington State and Alaska, (206) 882-8088. In Canada (416) 673-7638. Microsoft, the Microsoft logo and CodeView are registered trademarks and QuickC is a trademark of Microsoft Corporation. IBM is a registered trademark and Personal System/2 is a trademark of International Business Machines Corporation.

NOW SHIPPING

got everything else, and for \$99.95, it's all the compiler that even the most demanding Pascal programmer needs.

by Kent Porter

CodeView

Product:

Microsoft CodeView, Version 2.1

Target:

PC or PS/2 and compatibles

Requires:

DOS 2.0 or later

Price:

Comes with Microsoft MASM 5.0, C 4.0 or later, and FORTRAN

Vendor:

Microsoft, 16011 N.E. 36th Way,
P.O. Box 97017, Redmond, WA 98073;
(800) 426-9400

I can remember when the hardest thing about debugging code was trying to decide whether to print the whole link map or to just jot

down an item or two. Printing the whole thing could take 20 minutes or more. The other tough decision was whether or not to compile the code with listing output so that I could follow my program at the source level.

Thankfully, those days have gone. Symbolic debuggers did away with the need for the link map; source-level debuggers "knew" which source lines went with which machine instructions. But somehow, I still felt there was more a debugger could do for me—such as knowing the type as well as the address of variables, for example; or not intermixing output from the debugger with output from the program under test; or setting it to watch certain variables for me.

To address most of these issues, Microsoft created CodeView 1.0, which came bundled with Microsoft C, Version 4.0. Although memory was still a huge problem, at least the debugger knew as much as I did about the logistics of the program.

Version 2.1 relieves the memory problem by making use of expanded (EMS) memory. This is by no means a perfect solution, however.

Microsoft CodeView has two main operating modes. One is purely prompt/command-oriented and is appropriate for use on MS-DOS machines that do not support one of the conventional IBM display adapters. The second is a windowing mode, usable only on MS-DOS machines with compatible display adapters. I'll discuss the windowing mode here, but nearly all CodeView's power is likewise available in command mode.

CodeView is easy to use, but not at the expense of power. Most if not all of Symdeb's commands can be used in a dialog window. This scrolling window contains the customary prompt as well as output from issued commands. It even includes some history so you can go back and look at some output again.

There are several other windows that are much more interesting. First, there's the source window; in it, some portion of your source code, usually the portion being executed, is displayed. The source window has three modes—source only, assembly language only, and mixed source and assembly language. The currently executing line is in a color bar.

The remaining two windows are optional. The register window displays the current value of all the processor's registers (all 32 bits if the processor is an 80386 and you have set 386 mode in CodeView). The watch window displays the current value of arbitrary expressions, which allows you to monitor a variable or a more complicated expression.

You can set breakpoints to any given line simply by tapping F9. The desired line is displayed in high-intensity white, indicating a breakpoint. The dialog version of the breakpoint-set command has more power, allowing the setting of pass counts and commands to be executed upon breaking. F7 executes a given statement.

There are the usual two forms of

The Heap Expander™ version 2.0

Now your programs can have virtually unlimited heap space using expanded memory, extended memory, disk space, or any combination of the three. And it's all transparent. The Heap Expander's initialization code checks the system's resources and uses whatever is available.

still
\$59.95*

- Uses LIM-standard expanded memory if present.
- Uses AT-style extended memory if present.
- Swaps data to disk as needed.

Libraries and Source Code for:

- Turbo C
- Microsoft C 4.0 and 5.0
- Turbo Pascal 3.0 and 4.0

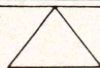
Requires an IBM PC, XT, AT, or close compatible with MS-DOS or PC-DOS version 2.0 or above

MC/VISA/COD call
1-800-248-1045 x 100 (US)
1-800-952-5560 x 100 (Idaho)

*Idaho residents add 5% sales tax
Foreign customers add \$4.00 for shipping and handling.

The Tool Makers

P.O. Box 8976
Moscow, Idaho 83843
208-883-4979



Why We're Betting a Million Lines of Code on the SAS/C™ Compiler.

At SAS Institute Inc., we've invested more than 10 years of research—and over a million lines of code—in the SAS® System, the world's leading data analysis software. So you can bet we left nothing to chance when we chose the C language for the next generation of our software.

We selected C for the portability it would bring to the SAS System, but weren't about to risk our code on just **any** mainframe C compiler. So we tried them all. When none could meet our exacting requirements, we created our own: the SAS/C compiler.

We Developed It.

Support It. Use It.

The SAS/C compiler set new standards for efficiency and technical quality, with:

- A source-level debugger that includes structure display, ABEND recovery, and debugger I/O exits for debugging specialized applications
- Reentrant object code
- Highly optimized generated code
- Use of standard IBM linkage conventions, with support for 31-bit addressing
- A CMS Rexx/TSO CLIST interface
- Support for signal handling including program checks and terminal interrupts, and non-standard signals such as timer interrupts and stack overflow
- Many built-in functions including string handling
- In-line assembler.

And when we combined these features with outstanding technical support and frequent updates—both provided free—software developers everywhere took notice. The SAS/C compiler is now the market leader, installed in hundreds of commercial firms and academic institutions.

Test It. Compare It.

FREE for 30 Days.

We're betting you've set the same high standards. That's why we'd like to send you the SAS/C compiler, under

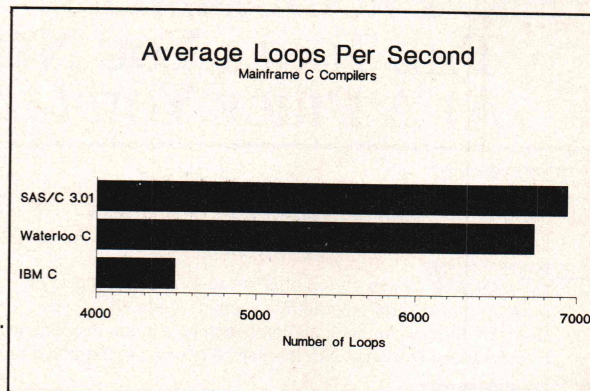
OS or CMS, for a free 30-day evaluation. We'll also send you a free copy of a leading benchmark program. Compare our compiler with any other. Odds are, you'll choose the SAS/C compiler.

Just mail the coupon below. Or call your Software Sales Representative at (919) 467-8000.



SAS Institute Inc.
SAS Circle ☐ Box 8000
Cary, NC 27512-8000
Phone (919) 467-8000
Fax (919) 469-3737

Using a C version of the Dhrystone benchmark, the latest SAS/C compiler release produces the fastest code among the top 3 mainframe compilers. It even tops our own previous release by 35%.



I'd like to put the SAS/C™ compiler to the test with a free 30-day trial, and my free copy of the Dhrystone benchmark program. Give me the details.

Please complete, or attach your business card.

Name _____ Title _____
Company _____
Address _____
City _____ State _____ ZIP _____
Telephone _____

Mail to: SAS Institute Inc., Attn: CC, SAS Circle, Box 8000,
Cary, NC, USA, 27512-8000

single-step—*STEP INTO* (F8) and *STEP OVER* (F10). *STEP INTO* steps into a function call being stepped, whereas *STEP OVER* executes until the stepped function returns. Both modes do exactly the same thing if there is no function call at the current location. Both are sensitive to the current source window mode if, in source-only mode, the single steps are source steps. In the other two modes, steps are single instructions.

One of the most useful additions to any debugger is the ability to use data breakpoints. Data breakpoints allow you to stop program execution when a particular variable, expression, or region of memory changes. Microsoft calls these tracepoints. Of course, you may not be interested in all changes but only in the one change that results in an erroneous value. If you can write an expression that yields zero when all is OK and nonzero when the program should stop, you can use watchpoints.

The main problem with data breakpoints is speed. Most processors don't implement any special mechanism for them, and as a result, a debugger must do a single step, check data breakpoints, single step, check, and so on. It's easy to see why this can be slow. But Microsoft has started to take advantage of the 80386. Tracepoints of limited range can be done while the processor is running at full speed on the 386 because of the processor's debug registers. Note that watchpoints do not take advantage of this feature, and I have no idea why not. To use this feature, you must give */r* on the CodeView command line, as documented in the readme file.

I like mice, so CodeView's mouse support is a big plus for me. Both F7 (go until) and F9 (set breakpoint) are implemented on the mouse buttons, so these operations are simply point and click. You can scroll both the source and dialog windows and you can move the line between them using the mouse.

One last feature is the ever-desirable *CONTINUE UNTIL RETURN*. If you have traced into a procedure and then wish to execute until the function returns, simply pull down the Calls menu and point to the function to which you wish to return. The source window then displays that function and the cursor is positioned such that pressing F7 executes the desired result. Note that this feature is available only in window mode, not in dialog mode.

A combination of excellent performance, richness of features, and a large installed base will probably guarantee that CodeView will remain a reference standard in debuggers for some time to come. And despite a few flaws, that's as it should be.

by Richard A. Relp

DDJ

C PROGRAMMERS! THE TOOLS YOU NEED AT A PRICE YOU'LL LIKE

BTree

Supports all index file operations. Very quick sequential or random access, duplicate keys, multiple indices, fixed and variable length data records are all supported.

75.00

ISAM

Works on top of BTree to provide a simple, yet powerful application program/file system interface. Complex filesystem manipulation becomes a snap. Provides the power of a database manager with the flexibility of a programming language.

40.00

lp

Finally, a completely device independent printer library! lp drives any printer as accurately as possible and allows easy access to its most sophisticated features. Multiple fonts, multi-column output, complex margin formatting, and much more. Pays for itself the first time it's used.

75.00

Snake

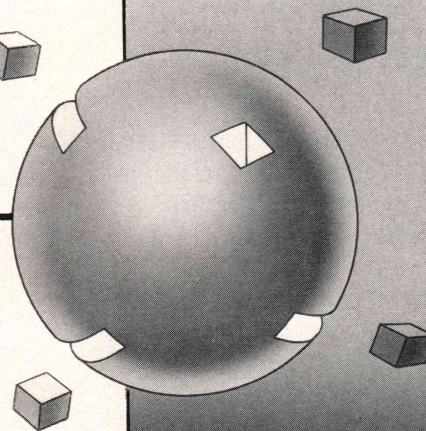
The ultimate 'make' utility. We couldn't find a good one, so we wrote a great one. Has all kinds of powerful features including wild card filename expansion, nested macros, and multiple dependency and rules definitions. Ready to go for MS-DOS; C source is there if you use another operating system.

59.00

Combine & Save: BTree + ISAM + lp **159.00** + Snake **199.00**
Each product includes a typeset manual, example programs, and complete C source code that runs on any operating system. Softfocus products may be incorporated into applications royalty-free.

Credit card orders accepted. Visa, M/C, Amex. Dealer inquiries invited.

**NOW
MULTI-
USER
AVAILABLE
60.00**

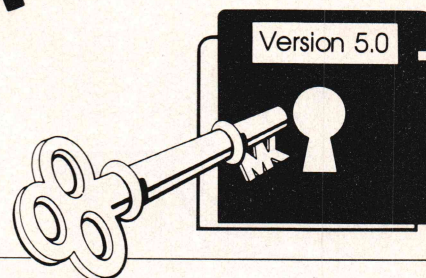


softfocus

1343 Stanbury Drive
Oakville, Ontario, Canada
L6L 2J5
(416) 825-0903

MASTER*KEY

Unlocks Everything!



turn this
into this!

```
C:\>DEBUG PROGRAM.COM
-D100 136
8848:0100 EB 18 49 6E 63 6F 72 72-65 63 74 20 44 4F 53 20 k.Incorrect DOS
8848:0110 76 65 72 73 69 6F 6E OD-0A 24 50 B4 30 CD 21 86 version..EP40M!
8848:0120 E0 3D 36 01 72 05 3D OA-02 76 09 BA 02 01 B4 09 ``6.r...v...4.
8848:0130 CD 21 CD 20 58 EB 2F MIM Xk/
-Q
```

MASTER*KEY No Other Product Comes Close!

An EXPERT may not know the solution, but always knows where to find it.

MASTER*KEY HELPS ANYONE solve those confusing and frustrating software puzzles more rapidly and easily than any other software available, at any cost! It gives you know-how within hours that may otherwise take years of experience. Create a new program from an old one. DON'T REINVENT THE WHEEL!

MASTER*KEY - Smart!

MASTER*KEY is an intelligent self-documenting MS-DOS reverse assembler. Its sophisticated procedures swiftly race through massive and baffling object code files to effortlessly discover potential trouble spots.

MASTER*KEY - Educational!

YOU DON'T NEED TO KNOW ASSEMBLY LANGUAGE! MASTER*KEY will take any program from your IBM-compatible computer and return fully-documented, easily-understood assembly language source code (Microsoft MASM 4.0 compatible).

MASTER*KEY - Easy To Use!

MASTER*KEY works both automatically from the DOS command line or interactively from menus similar to Lotus Corporation's 1-2-3 or Symphony. No need to remember any new commands or continually refer to a manual. Use it immediately!

Minimum System Requirements:

256K + 8088/8086/80186/80286/80386 PC
MS-DOS or PC-DOS 2.0 +
One 360K DSDD Floppy Drive (IBM PC Format)

MS-DOS is a trademark of Microsoft.
PC-DOS is a trademark of IBM.

```
H00100: JMP      Short H0011A                      ;00100 EB18      --
;-----
      DB      "Incorrect DOS version"              ;00102 496E636F727265
      DB      ODh                                     ;00117
      DB      0Ah                                     ;00118
      DB      "g"                                    ;00119 24
;-----
H0011A: PUSH     AX                                ;0011A 50          P
      MOV     AH,30h                               ;0011B B430        _O
      INT     21h                                   ;0011D CD21        _!
      XCHG    AH,AL                                ;0011F 86E0        _
      CMP     AX,0136h                             ;00121 3D3601      =6
      JB      H0012B                               ;00124 7205        r_
      CMP     AX,020Ah                             ;00126 3D0A02      r_
      JBE     H00134                               ;00129 7609        v_
H0012B: MOV     DX,0102h                           ;0012B BA0201      ---
      MOV     AH,09h                               ;0012E B409        ---
      INT     21h                                   ;00130 CD21        _!
      INT     20h                                   ;00132 CD20        _
;-----
H00134: POP      AX                                ;00134 58          X
      JMP     Short H00166                          ;00135 EB2F        _/
;-----
```

MASTER*KEY XREF - PROGRAM.XRF

Page 1

```
0102h      : 121 2F5 301 320
020Ah      : 126
03CBh      : 12B
1-Display_String : 130 591 610
1-DOS_Ver_Number : 11D
H00100      : 100
H0011A      : 100 11A
H0012B      : 124 12B
H00134      : 129 134
H00166      : 135
TERM_normally:20h : 132
```

NOTE: The cross-reference is by memory location within the program file!

NOTE: The output is totally Microsoft MASM-compatible.

(not copy protected)

MASTER*KEY will guide you step by step to:

1. Help you learn assembly language, if you desire.
2. Discover how any program runs or why it doesn't.
3. Alter or remove unwanted object code from any program.
4. Incorporate routines from compiled programs into other assembly language, Basic, C, or Pascal programs.
5. Make software more compatible with your computer. Be certain a questionable program won't damage your system BEFORE you run it.
6. Modify software to operate with other versions of DOS.
7. Customize your COMMAND.COM or other executable program directly or by reassembling your altered MASTER*KEY source code.

Order Now!
Just \$79⁹⁵

Phone orders accepted on MC or VISA

\$82.45 (includes \$2.50 shipping)

\$87.65 in California (includes tax & shipping) C.O.D. orders add \$2.00

(714) 596-0070

Please send MASTER*KEY!

Send checks to:

Sharpe Systems Corporation

2320 E Street, Dept. 44, La Verne, CA 91750

Name _____

Address _____

City _____

State _____

Zip _____

Dealer/Distributor Inquiries Welcome

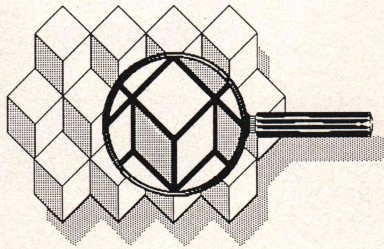
Sharpe Systems Corporation

2320 E Street, Dept. 44, La Verne, CA 91750 714-596-0070

MASTER*KEY should not be confused with any public domain or share ware software that may have a similar name or be a similar product.

CIRCLE NO. 193 ON READER SERVICE CARD

OF INTEREST



Object-Oriented Programming

Productivity Products International is offering an object-oriented software engineering environment that contains Objective-C, Vici, and a catalog of Software-ICs. Objective-C is an object-oriented software engineering language that includes such features as encapsulation, which holds data within an object and surrounds the data with a shell of procedures; inheritance, which builds and reuses code; and dynamic binding, which allows Objective-C objects to decide at run time which routine to run. Vici is a prototyping and debugging environment. Software-ICs are modules of reusable code that give programmers the same advantages as integrated circuits give hardware designers.

The PPI product set is available for the DEC VAX, Sun Microsystems, HP-9000, and Apollo lines and the IBM PC AT. Source code licenses are available for users who want to port the environment to their own proprietary hardware. Reader Service No. 16.

Productivity Products International Inc.

27 Glen Rd.

Sandy Hook, CT 06482

(203) 426-1875

A new programming product for the Macintosh is available from **Coral Software**. Object Logo features object-oriented programming tools, an interpreter for interactive debugging, a compiler that automatically translates each procedure into native code, a user interface that conforms

to the Macintosh standards, complex and rational arithmetic, arrays, Macintosh ROM support, and a variety of advanced primitives and debugging tools. Object Logo is available for the Macintosh 512, 512E, Plus, and SE and costs \$79.95. Reader Service No. 17.

Coral Software

P.O. Box 307

Cambridge, MA 02142

(617) 547-2662

Languages

A Modula-2 language systems is now available from **ana-systems**. Modula-2/68 is suitable for professional software development or academic instruction on systems using the Motorola line of 32-bit microprocessors. With Modula-2/68, program modules can be compiled separately. An executable process can be built by linking with previously compiled program modules or by linking with library procedures written in C. Modula-2 programs can easily access standard C object libraries, which means that modules can use previously written and debugged subroutine libraries written in other programming languages. For special applications based on the MC68000 processor line, the linked Modula-2 programs can be written out in the Motorola S-record format and then easily downloaded to standard development systems.

Modula-2/68 is available for the Convergent Technologies line of Unix systems and sells for \$1,200. Reader Service No. 18.

ana-systems

697 Saturn Ct.

P.O. Box 4759

Foster City, CA 94404-0759

(415) 341-1768

WATCOM has announced two new C language products for IBM PC and PS/2 DOS systems. WATCOM C 6.0 is an optimizing compiler that comes with a full range of programming tools, including a windowed source-level debugger. WATCOM Express C provides an integrated development environment, including an editor, compiler, debugger, and run-time library that are all memory-resident.

Programs may be compiled in memory and then executed directly without separate link and load steps. Reader Service No. 19.

WATCOM Products Inc.

415 Phillip St.

Waterloo, Ont.

Canada N2L 3X2

(519) 886-3700

C Workshop from **Wordcraft** is a program that teaches C. It also contains a programming environment that users can use for experimenting, prototyping, and developing the modules for any programming project. C Workshop contains a full standard K & R compiler, an editor, a run-time library, tutorial software, and a book. The software uses 220K and runs on 320K MS-DOS computers with industry-standard BIOS. The package sells for \$69.95. Reader Service No. 20.

Wordcraft

3827 Penniman Ave.

Oakland, CA 94619

(800) 227-2400

PL/D from **Dair Computer Systems** is a new system language that allows users the speed and control of assembly language with the ease of expression of a compiler. Like assembly language, PL/D has no run-time library overhead. PL/D code is inherently capable of relocation for EPROM-based embedded control, a common PL/D application. Features of the language include macro and conditional compile facilities, library functions that can be incorporated into the program at compile time as SYS source files, 17 options that can be specified within the source, and some features similar to Forth and C. Source code of the compiler is available. PL/D requires a 192K IBM PC AT or PC/XT with MS-DOS 2.0 or later and sells for \$124.95. Reader Service No. 21.

Dair Computer Systems

3440 Kenneth Dr.

Palo Alto, CA 94303

(415) 494-7081

Fun Stuff

If you are tired of programming and want to get out of the house, **Ex-**

panded Entertainment has released a new film that compiles some of the top shorts in computer animation in The Computer Animation Show. The show is a collection of 3-D character animation, abstract work, corporate show reels, experimental animation, and several ground-breaking music videos. The show is presented in 35 millimeter and charts the growth of computer graphics over the last five years. The movie debuts in 400 cities nationwide throughout 1988. Reader Service No. 22.

Expanded Entertainment
2222 S. Barrington Ave.
Los Angeles, CA 90064
(213) 473-6701

Hardware

RasterOps Corp. has introduced a high-resolution color graphics board for the Macintosh II. The ColorBoard 1/104 features a single-slot design, true color capabilities, and 1,024×768 pixels on a 24-bit color plane capable of displaying 16.7 million colors simultaneously. The Color-

Board 1/104 is priced at \$2,795 and runs on the Macintosh II. Reader Service No. 23.
RasterOps Corp.
10161 Bubbl Rd.
Cupertino, CA 95014
(408) 446-4090

The EVERCOM II 24 is a 2,400-bps modem for the IBM PS/2 with Micro Channel architecture. Available from **Everex**, this modem features state-of-the-art signal processing and adaptive equalization, auto-dial, auto-answer, auto-speed-matching to the calling modem, and auto-sense of tone/pulse dialing. Users can easily switch from voice to data and data to voice during a call. Built-in intelligence can detect and respond to such variables as transmission speed and data format. International compatibility is supplied by incorporating Hayes, Bell 212A/103, CCITT V.22, and V.22 bis protocol standards. BitCom, a menu-driven communications program is also included. The EVERCOM II 24 is priced at \$299. Reader Service No. 24.

Everex
48431 Milmont Dr.
Fremont, CA 94538
(415) 498-1111

DigiBoard has introduced the OpenEnde, a multichannel board for the PS/2. The OpenEnde features DigiBoard's modular I/O concept, the I/O Mate. The I/O Mate contains I/O components and mounts to the OpenEnde portion of the board that holds the processor and memory. I/O Mates are available in both 4- and 8-port configurations with an added synchronous channel available on the I/O Mate+ series. The OpenEnde features an on-board 80186 microprocessor operating at 12 MHz, 256K of dual-ported RAM, and up to 64K of ROM to store user-defined programs. The OpenEnde is compatible with MS-DOS and Xenix and will support other operating systems. List prices are \$1,349 with I/O Mate-8, \$1,399 with I/O Mate8+, \$1,149 with I/O Mate-4, and \$1,199 for I/O Mate4+. Reader Service No. 25.

"Dick Johnson in accounting is having a heart attack!"

Would you know what to do?
Would anyone in your company be
able to help?

One of your employees is stricken.
Breathing and heartbeat have stopped.
Does anyone know what to do until help
arrives?

The American Red Cross can
train your employees in CPR—Cardio-
pulmonary Resuscitation, a first aid
method that sustains life.

It's just one of the ways the
Red Cross helps you keep your company
healthy and safe.

Contact your local Red Cross
Chapter and ask about CPR training.
That way, when disaster strikes, you can
all breathe a little easier.



American Red Cross



Announcing WKS LIBRARY

NEW!

The Lotus "Wrap-Around" for C Programs

Now you can write and read Lotus worksheets
directly from a C program!

WKS LIBRARY lets you:

- avoid time-consuming file translation steps
- control the execution of 1-2-3 from inside your application
- use Lotus as a data entry screen in your C program
- generate "live" financial statements with formulas for totals

Feature this:

- reads and writes .WKS, .WK1 and .WR1 worksheets
- writes integers, floats, strings, formulas, macros and dates using *wprintf()*
- reads using *wscanf()*, converting column contents to C variables according to format specs
- has low-level functions for manipulating individual cells
- provides Lotus control functions such as: formats, column widths, initial cursor position, range names, cell protection
- supports Lattice, Microsoft & Turbo compilers for DOS, plus most UNIX environments

Source Code provided

No Royalties on executable programs

Only \$89

(includes free
800-line support)



ORDER TODAY!

**Toll-free
(800) 367-9882**



Tenon Software, Inc.

1980 - 112th NE, #250, Bellevue, WA 98004
(206) 453-1914 (in Washington state)

OF INTEREST

(continued from page 131)

DigiBoard Inc.
6751 Oxford St.
St. Louis Park, MN 55426
(612) 922-8055

Kinetic Access is a security product from **Kinetic Corp.** that offers micro system security through an easy-to-use program. The total system includes a hardware device that controls the booting process and a resident control program that requires 45K RAM while in operation. The hardware device can be either an EPROM, which fits into an available ROM socket, or a short expansion card that plugs into any available slot on the PC. After installation of Kinetic Access, every time the system is booted up, Kinetic Access password protects users' files and applications. Kinetic Access is available for users of IBM PC, XT, AT, and true compatibles operating with MS-DOS, Versions 2.x or 3.x. The product is priced at \$129.95 per unit with EPROM and \$149.95 per unit with short expansion board. Reader Service No. 26.

Kinetic Corp.
Distillery Commons 240
Lexington Rd. @ Payne
Louisville, KY 40206-1990
(502) 583-1679

Tools and Utilities

A data-compression package has been released by **Isogon Corp.** NEWS-SPACE is a RAM-resident utility that significantly increases the storage capacity of any IBM or IBM-compatible PC hard disk. The program automatically and transparently compresses word processing, spreadsheet, database, and all other kinds of data files without any user involvement. Although the kind of compression achieved depends on the nature of the data, 50 percent overall compression is average. NEWS-SPACE operates with all application programs and RAM-resident utilities and works on all PC-DOS and MS-DOS machines running Version 2.0 or later, including the PS/2. The program sells for \$69.95. Reader Service No. 27.

Isogon Corp.

330 Seventh Ave.
New York, NY 10001
(212) 967-2424

Quinn-Curtis has announced Science and Engineering Tools for the Macintosh 512K or bigger. The package includes procedures for general statistics, multiple regression, curve fitting, integration, FFTs, solving differential and simultaneous equations, matrix math, complex math, data smoothing, linear programming, root finding, and special functions. A 160-page manual describes the form, function, and parameters of every procedure and function. All the software tools are supplied on a 3.5-inch disk in both Lightspeed Pascal and Turbo Pascal source code. The Science and Engineering Tools package retails for \$74.95. Reader Service No. 28.

Quinn-Curtis
49 Highland Ave.
Needham, MA 02194
(617) 444-7721

DDJ

ADD TO THE POWER OF YOUR PROGRAMS WHILE YOU SAVE TIME AND MONEY!

CBTREE does it all! Your best value in a B+tree source!

Save programming time and effort.

You can develop exciting file access programs quickly and easily because CBTREE provides a simple but powerful program interface to all B+tree operations. Every aspect of CBTREE is covered thoroughly in the 80 page Users Manual with complete examples. Sample programs are provided on disk.

Gain flexibility in designing your applications.

CBTREE lets you use multiple keys, variable key lengths, concatenated keys, and any data record size and record length. You can customize the B+tree parameters using utilities provided.

Your programs will be using the most efficient searching techniques. CBTREE provides the fastest keyed file access performance, with multiple indexes in a single file and crash recovery utilities. CBTREE is a full function implementation of the industry standard B+tree access method and is proven in applications since 1984.

Access any record or group of records by:

- Get first
- Get previous
- Get less than
- Get greater than
- Get sequential block
- Get all partial matches
- Insert key and record
- Delete key and record
- Change record location
- Get last
- Get next
- Get less than or equal
- Get greater than or equal
- Get partial key match
- Get all keys and locations
- Insert key
- Delete key

Increase your implementation productivity.

CBTREE is over 8,000 lines of tightly written, commented C source code. The driver module is only 20K and links into your programs.

Port your applications to other machine environments.

The C source code that you receive can be compiled on all popular C compilers for the IBM PC and also under Unix, Xenix, and AmigaDos! No royalties on your applications that use CBTREE. CBTREE supports multi-user and network applications.

CBTREE IS TROUBLE-FREE, BUT IF YOU NEED HELP WE PROVIDE FREE PHONE SUPPORT.

ONE CALL GETS YOU THE ANSWER TO ANY QUESTION!

CBTREE compares favorably with other software selling at 2,3 and 4 times our price.

Sold on unconditional money-back guarantee.

YOU PAY ONLY \$159 - A MONEY-SAVING PRICE!

TO ORDER OR FOR ADDITIONAL INFORMATION

CALL 1-800-346-8038 or (703) 847-1743

OR WRITE

NOW! Variable length records.

NEW! --- Limited Time Offer.
Object Library for Only \$49!



PEACOCK SYSTEMS, INC.

Peacock Systems, Inc., 2108-C Gallows Road, Vienna, VA 22180

C CODE FOR THE PC

source code, of course

C Source Code

Bluestreak Plus Communications (two ports, programmer's interface, terminal emulation)	\$400
CQL Query System (SQL retrievals plus windows)	\$325
GraphiC 4.1 (high-resolution, DISSPLA-style scientific plots in color & hardcopy)	\$325
Barcode Generator (specify Code 39 (alphanumeric), Interleaved 2 of 5 (numeric), or UPC)	\$300
Greenleaf Data Windows (windows, menus, data entry, interactive form design)	\$295
Vitamin C (MacWindows)	\$200
resident C (TSRify C programs, DOS shared libraries)	\$165
Essential C Utility Library (400 useful C functions)	\$160
Essential Communications Library (C functions for RS-232-based communication systems)	\$160
Greenleaf Communications Library (interrupt mode, modem control, XON-XOFF)	\$150
Greenleaf Functions (296 useful C functions, all DOS services)	\$150
OS/88 (U**x-like O/S, many tools, cross-development from MS-DOS)	\$150
Turbo G Graphics Library (all popular adapters, hidden line removal)	\$135
CBTree (B+tree ISAM driver, multiple variable-length keys)	\$115
MultiDOS Plus (DOS-based multitasking, intertask messaging, semaphores)	\$115
PC/IP (CMU/MIT TCP/IP implementation for PCs)	\$100
B-Tree Library & ISAM Driver (file system utilities by Softfocus)	\$100
The Profiler (program execution profile tool)	\$100
Entelekon C Function Library (screen, graphics, keyboard, string, printer, etc.)	\$100
Entelekon Power Windows (menus, overlays, messages, alarms, file handling, etc.)	\$100
Wendin O/S Construction Kit or PCNX, PCVMS O/S Shells	\$95
QC88 C Compiler (ASM output, small model, no longs, floats or bit fields, 80+ function library)	\$90
JATE Async Terminal Emulator (includes file transfer and menu subsystem)	\$80
MultiDOS Plus (DOS-based multitasking, intertask messaging, semaphores)	\$80
ME (programmer's editor with C-like macro language by Magma Software)	\$75
WKS Library (C program interface to Lotus 1-2-3 program & files)	\$65
Quincy (interactive C interpreter)	\$60
EZ_ASM (assembly language macros bridging C and MASM)	\$60
PTree (parse tree management)	\$60
HELP (pop-up help system builder)	\$50
Multi-User BBS (chat, mail, menus, sysop displays; uses Galacticom modem card)	\$50
Heap Expander (dynamic memory manager for expanded memory)	\$50
Make (macros, all languages, built-in rules)	\$50
Vector-to-Raster Conversion (stroke letters & Tektronix 4010 codes to bitmaps)	\$50
Coder's Prolog (inference engine for use with C programs)	\$45
C-Help (pop-up help for C programmers ... add your own notes)	\$40
Biggerstaff's System Tools (multi-tasking window manager kit)	\$40
CLIPS (rule-based expert system generator, Version 4.0)	\$35
TELE Kernel (Ken Berry's multi-tasking kernel)	\$30
TELE Windows (Ken Berry's window package)	\$30
Clisp (Lisp interpreter with extensive internals documentation)	\$30
Translate Rules to C (YACC-like function generator for rule-based systems)	\$30
6-Pack of Editors (six public domain editors for use, study & hacking)	\$30
ICON (string and list processing language, Version 6 and update)	\$25
LEX (lexical analyzer generator)	\$25
Bison & PREP (YACC workalike parser generator & attribute grammar preprocessor)	\$25
AutoTrace (program tracer and memory trasher catcher)	\$25
C Compiler Torture Test (checks a C compiler against K & R)	\$20
Benchmark Package (C compiler, PC hardware, and Unix system)	\$20
TN3270 (remote login to IBM VM/CMS as a 3270 terminal on a 3274 controller)	\$20
A68 (68000 cross-assembler)	\$20
List-Pac (C functions for lists, stacks, and queues)	\$20
XLT Macro Processor (general purpose text translator)	\$20
C Tools (exception macros, wc, pp, roff, grep, printf, hash, declare, banner, Pascal-to-C)	\$15

Data

WordCruncher (text retrieval & document analysis program)	\$275
DNA Sequences (GenBank 48.0 of 10,913 sequences with fast similarity search program)	\$150
Protein Sequences (5,415 sequences, 1,302,966 residuals, with similarity search program)	\$60
Webster's Second Dictionary (234,932 words)	\$60
U. S. Cities (names & longitude/latitude of 32,000 U.S. cities and 6,000 state boundary points)	\$35
The World Digitized (100,000 longitude/latitude of world country boundaries)	\$30
KST Fonts (13,200 characters in 139 mixed fonts: specify TeX or bitmap format)	\$30
USNO Floppy Almanac (high-precision moon, sun, planet & star positions)	\$20
NBS Hershey Fonts (1,377 stroke characters in 14 fonts)	\$15
U. S. Map (15,701 points of state boundaries)	\$15

The Austin Code Works

11100 Leafwood Lane

Austin, Texas 78750-3409 USA

acw!info@uunet.uu.net

Voice: (512) 258-0785

BBS: (512) 258-8831

FidoNet: 1:382/12

Free surface shipping on prepaid orders

CIRCLE NO. 198 ON READER SERVICE CARD

MasterCard/VISA

Upgrade Your Technology

We're Programmer's Connection, the leading independent dealer of quality programmer's development tools for IBM personal computers and compatibles. We can help you upgrade your programming technology with some of the best software tools available.

Comprehensive Buyer's Guide. The CONNECTION, our new Buyers Guide, contains prices and up-to-date descriptions of over 600 programmer's development tools by over 200 manufacturers. Each description covers major product features as well as special requirements, version numbers, diskette sizes, and guarantees.

How to Get Your FREE Copy: 1) Use the reader service card provided by this journal; 2) Mail us a card or letter with your name and address; or 3) Call one of our convenient toll free telephone numbers.

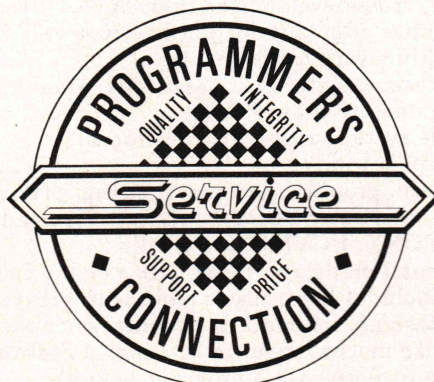
If you haven't yet received your copy of the Programmer's Connection Buyer's Guide, act now. Upgrading your programming technology could be one of the wisest and most profitable decisions you'll ever make.

USA 800-336-1166

Canada 800-225-1166
Ohio & Alaska (Collect) 216-494-3781
International 216-494-3781
TELEX 9102406879

Business Hours: 8:30 AM to 8:00 PM EST Monday through Friday
Prices, Terms and Conditions are subject to change.
Copyright 1988 Programmer's Connection Incorporated

Sale Prices effective through 03/31/88



386 products		List	Ours
386 ASM/386 LINK Cross Asm by Phar Lap		495	389
386 DEBUG Cross Debugger by Phar Lap		195	129
NDP C-386 by MicroWay	New	595	529
NDP ForTran-386 by MicroWay	New	595	529
PC-MOS/386 Single-User by The Software Link		195	155
PC-MOS/386 S-Users by The Software Link		595	539
PC-MOS/386 25-Users by The Software Link		995	869

OPTASM

List \$195 Ours \$179 Sale \$159

OPTASM is a high performance macro assembler which is compatible with the Microsoft Macro Assembler 5.0 (except for 80386 support). It provides many advanced features as well as speed that's four times faster than MASM. And utilizing the built-in make utility can increase this speed to 10 times! OPTASM eliminates most of the problems with using MASM. It generates correct code in all forward reference cases that cause phase errors in MASM. OPTASM is one of the fastest MASM-compatible assemblers available.

aldebaran products

Source Print by Aldebaran Labs	97	74
Tree Diagrammer by Aldebaran Labs	77	59

alsys products

Ada GSA-validated w/maintenance	3355	3119
Ada Developer's Toolset Volumes 1 & 2	New 995	919
AdaQUERY	New 200	185

assembly language

Cross Assemblers Various by 2500 AD	CALL	CALL
DMS Resident-ASM by American Software Intl	150	139
OPTASM by SLR Systems	New, Sale 195	159
risC by IMSI	New, Sale 80	59

blaise products

ASYNCH MANAGER Specify C or Pascal	175	135
C TOOLS PLUS/5.0	129	99
KeyPlayer Super Batch Program	50	45
LIGHT TOOLS for Datatight C	Sale 100	55
PASCAL TOOLS/TOOLS 2	175	135
RUNOFF Text Formatter	50	45
Turbo ASYNCH PLUS/4.0	129	99
Turbo C TOOLS	129	99
Turbo POWER TOOLS PLUS/4.0	129	99
VIEW MANAGER Specify C or Pascal	275	199

borland products

EUREKA Equation Solver	167	105
Paradox 1.1 by Ansa/Borland	495	359
Paradox 2.0 by Ansa/Borland	725	525
Paradox Network Pack by Ansa/Borland	995	725
Quattro: The Professional Spreadsheet	New 195	125
Reflex: The Analyst	150	99
Sidekick	85	57

Superkey	100	64
Turbo Basic Compiler	100	64
Turbo Basic Database Toolbox	100	64
Turbo Basic Editor Toolbox	100	64
Turbo Basic Telecom Toolbox	100	64
Turbo C Compiler (Call for support products)	100	64
Turbo Lighting	100	64
Turbo Lightning Word Wizard	70	47
Turbo Pascal	Sale 100	59
Turbo Pascal Database Toolbox	100	64
Turbo Pascal Developer's Toolkit	395	259
Turbo Pascal Editor Toolbox	100	64
Turbo Pascal Gameworks Toolbox	100	64
Turbo Pascal Graphics Toolbox	100	64
Turbo Pascal Numerical Methods Toolbox	100	64
Turbo Pascal Tutor	70	41
Turbo Prolog Compiler	100	64
Turbo Prolog Toolbox	100	64

c language

Eco-C88 Modeling Compiler by Ecosoft	Sale 100	69
Lattice C Compiler vers. 3.2 from Lattice	500	265
Optimum-C by Datatight	139	95

Peabody Pop-Up Reference Utility by Copia International

List \$100 Ours \$89

Peabody is a fast and flexible on-line reference utility with databases available for Turbo Pascal or Microsoft C. It provides instant, accurate and complete language information in pop-up frames at the touch of a key. With Peabody, you can select general topics from a structured subject menu, or use Peabody's hyperkey to get instant help for the keyword closest to the cursor. Specify database desired. Additional databases are available for \$100 with manual or \$50 without manual.

c utilities

C++ by Guidelines	New Version 295	259
C Programmer's Toolbox I by MMC AD	New 80	69
C Programmer's Toolbox II by MMC AD	New 80	69
C Programmer's Toolbox Combo by MMC AD	New 130	115
c-scape by Oakland Group	New 279	265
c-tree & r-tree Combo by FairCom	650	519
c-tree ISAM File Manager	395	315
r-tree Report Generator	295	239
CBTREE by Peacock Systems	New, Sale 159	119
CQL by Machine Independent Software	New 395	329
Curses Window Dev Pkg by Aspen Scientific	Sale 119	95
with Source Code	289	235
dBx dBASE to C Translator by Desktop AI	350	299
with Source Code	550	419
DMS Resident-C by American Software Intl	150	139
Entelekon Combo Package	200	99
Flash-up by Software Bottling	89	79
HALO Graphics by Media Cybernetics	300	205
HALO Development Pkg for Microsoft	595	389

ORDERING INFORMATION

FREE SHIPPING. Orders within the USA (including Alaska & Hawaii) are shipped FREE via UPS. Call for express shipping rates.

NO CREDIT CARD CHARGE. VISA, MasterCard and Discover Card are accepted at no extra cost. Your card is charged when your order is shipped. Mail orders please include expiration date and authorized signature.

NO COD OR PO FEE. CODs and Purchase Orders are accepted at no extra cost. No personal checks are accepted on COD orders. POs with net 30-day terms (with initial minimum order of \$100) are available to qualified US accounts only.

NO SALES TAX. Orders outside of Ohio are not charged sales tax. Ohio customers please add 5% Ohio tax or provide proof of tax-exemption.

30-DAY GUARANTEE. Most of our products come with a 30-day documentation evaluation period or a 30-day return guarantee. Please note that some manufacturers restrict us from offering guarantees on their products. Call for more information.

SOUND ADVICE. Our knowledgeable technical staff can answer technical questions, assist in comparing products and send you detailed product information tailored to your needs.

INTERNATIONAL ORDERS. Shipping charges for International and Canadian orders are based on the shipping carrier's standard rate. Since rates vary between carriers, please call or write for the exact cost. International orders (except Canada), please include an additional \$10 for export preparation. All payments must be made with US funds drawn on a US bank. Please include your telephone number when ordering by mail. Due to government regulations, we cannot ship to all countries.

MAIL ORDERS. Please include your telephone number on all mail orders. Be sure to specify computer, operating system, diskette size, and any applicable compiler or hardware interface(s). Send mail orders to:

Programmer's Connection
Order Processing Department
7249 Whipple Ave NW
North Canton, OH 44720

PANEL by Roundhill, Specify QuickC or Turbo C	129	95
PANEL Plus by Roundhill	495	395
RTC PLUS Fortran to C by Cobalt Blue	450	369
Sapiens V8 Virtual Memory Manager	300	265
TE Developer's Kit by Sub Systems	95	85
Vitamin C by Creative Programming	225	149
VC Screen Forms Designer	100	79
WKS LIBRARY by Tenon Software	89	79

database management

Clipper by Nantucket	695	379
dBASE III Plus by Ashton-Tate	695	389
dB2c FILES by Software Connection	New 199	179
dB2c TOOLKIT by Software Connection	New 299	249
dB2c WINDOWS by Software Connection	New 99	89
dFLOW by Wallsoft	149	119
Fox Base Plus by Fox Software	395	249
Genifer by Bytel	395	249
MAGIC PC by AKER	199	179
Q&A by Symantec	349	219
R-Base 5000 by Micromin	495	359
R-Base System V by Micromin	700	439
UI Programmer by Wallsoft	295	239

ASMLIB by BCSoft

List \$149 Ours \$125 Sale \$99

ASMLIB contains over 210 functions all easily accessible by a single CALL from within your assembler or high level language program. It features: access to keyboard, screen, disk files, EGA, CGA & Hercules adapters; windowing with overlapping windows; math and trig functions for four different data types; 8087 support, dynamic memory management; virtual disk file handling; and much more. Includes complete source code. Supports Microsoft C, Fortran, MASM, and Pascal.

debuggers

Periscope I with Board by Periscope	345	275
Periscope II with NMI Breakout Switch	175	139
Periscope II-X Software only	145	105
Periscope III 8 MHz version	995	795
Periscope III 10 MHz version	1095	875

digital products

Smalltalk/V by Digital	100	84
EGA/VGA Color Option	50	45
Goodies Diskette #1	50	45

Goodies Diskette #2	New	50	45
Goodies Diskette #3	New	50	45
Smalltalk/Comm		50	45

dos utilities

Desqview from Quarterdeck		130	115
Mace Utilities Paul Mace Software		99	85
XO-SHELL by Wyte Corporation	New	49	45

essential products

Breakout Debugger by Essential Software		125	89
C Utility Library		185	118
Essential Communications		185	118
Essential Communications with Break Out		250	189
Essential Graphics		250	183
/*resident C*/		99	85
with Source Code		198	148
ScreenStar		99	85
with Library Source Code		198	154

fortran language

FORTLIB by Sutrast	Sale	125	99
GRAFLIB by Sutrast	Sale	175	139
HALO Graphics by Media Cybernetics		300	205
INGRAF by Sutrast	New, Sale	250	199
Numerical Analyst by MAGUS		295	199
PLOTHI by Sutrast	Sale	175	139
PLOTHP by Sutrast	Sale	175	139

gimpel products

C-terp by Gimpel, Specify compiler	Sale	298	199
for UNIX/XENIX	Sale	498	299
PC Lint by Gimpel Software		139	99
Turbo C-terp for Turbo C by Gimpel	New, Sale	139	99

Flash-Up with FREE Mouse from Software Bottling of NY

List \$89 Ours \$79

Flash-Up is a memory-resident macro, menu and note maker compatible with most languages. Easy-to-use features include a pull-down interface and on-line help. And until 03/31/88, you'll also get a Microsoft compatible mouse FREE.

golden bow products

Vcache		50	47
Vfeature		80	74
Vfeature Deluxe		120	111
Vopt		50	47

greenleaf products

Greenleaf C Sampler specify QuickC or Turbo C		95	69
Greenleaf Comm Library		185	125
Greenleaf Data Windows Library		225	155
with Source Code		395	249
Greenleaf Functions		185	125

komputerwerk products

Finally by Komputerwerk		99	85
Finally Modules by Komputerwerk	New	99	85
Finally XGraf by Komputerwerk	New	99	85

lattice products

Lattice C Compiler ver 3.2 from Lattice		500	265
with Library Source Code		900	495
C Cross Reference Generator		50	37
C-Food Smorgasbord Function Library		150	95
with Source Code		300	179
C-Sprite Source Level Debugger		175	119
Curses Screen Manager		125	85
with Source Code		250	169
dBC III		250	159
with Source Code		500	318
dBC III Plus		750	594
with Source Code		1500	1184
LMK Make Facility	New Version	195	138
RPG II Combo All four items below	New Version	1400	1099
RPG II Compiler No Royalties	New Version	750	625
Screen Design Aid Utility for RPG II		350	309
SEU Source Entry Utility		250	199
Sort/Merge		250	199
Text Management Utilities		120	88

logitech products

LOGIMOUSE BUS with PLUS Pkg by LOGITECH		119	98
with PLUS & PC Paintbrush		149	119
with PLUS & CAD Software		189	153
with PLUS & CAD & Paint		219	179
with PLUS & First Publisher		179	139
LOGIMOUSE C7 with PLUS Package		119	98
with PLUS & PC Paintbrush		149	119
with PLUS & CAD Software		189	153
with PLUS & CAD & Paint		219	179
with PLUS & First Publisher		179	139
LOGITECH Modula-2 Development System		249	199
Modula-2 Compiler Pack		99	75
Modula-2 Toolkit		169	139

meridian products

AdaVantage GSA-validated	Sale	795	675
with Optimizer	New, Sale	995	879
AdaVantage Debugger	New	495	449
AdaVantage DOS Environment Package		50	47

AdaVantage Utility Packages		50	47
-----------------------------------	--	----	----

metagraphics products

MetaWINDOW No Royalties		195	159
MetaWINDOW/PLUS		275	229
QuickWINDOW/C for Microsoft QuickC		95	79
TurboWINDOW/C for Borland Turbo C		95	79
TurboWINDOW/Pascal for Borland Turbo Pascal		95	79

microcompatibles products

GRAFATIC		135	117
OMNILOT		295	265
Screen Only		195	175
PLOTMATIC		135	117
PRINTMATIC		135	117

microport products

DOSMerge286 Specify 2-Users or Unlimited		149	129
DOSMerge386 2-Users		395	345
DOSMerge386 Unlimited Users		495	429
System V/386 Combination		799	669
386 Runtime System		199	169
386 Software Development System		499	429
Text Preparation System		199	169
System V/AT Combination		549	465
AT Runtime System		199	169
AT Software Development System		249	209
Text Preparation System		199	169

AdaVantage Compiler by Meridian Software

List \$795 Ours \$735

with Optimizer:

List \$995 Ours \$879

The AdaVantage Compiler is a fast, efficient, production-quality Ada programming capability fully validated by the U.S. Department of Defense. It generates native 8086 code in Intel standard object format. Includes: compiler, linker, library mgmt. tools, support packages, runtime libraries, and a configuration tool. The optional Optimizer offers code improvements ranging from local optimizations to global subprogram removal.

microsoft products

Microsoft BASIC Compiler for XENIX		695	439
Microsoft BASIC Interpreter for XENIX		350	219
Microsoft C Compiler 5 w/CodeView		450	285
Microsoft COBOL Compiler with COBOL Tools		700	439
for XENIX		995	639
Microsoft Excel		495	319
Microsoft FORTRAN Optimizing Compiler		450	285
Microsoft FORTRAN for XENIX		695	439
Microsoft MACH 20	New	495	329
Microsoft Macro Assembler		150	99
Microsoft Mouse with Paint & Mouse Menus		150	99
with Microsoft Windows & Paint		200	139
with EasyCAD		175	119
Microsoft Pascal Compiler		300	189
for XENIX		695	439
Microsoft QuickBASIC		99	66
Microsoft QuickC		99	66
Microsoft Windows		99	66
Microsoft Windows 386		195	129
Microsoft Windows Development Kit		500	299
Microsoft Word		450	285
Microsoft Works		195	129

mks products

MKS AWK		75	65
MKS RCS Revision Control System		189	155
MKS Toolkit with MKS VI Editor		139	109
MKS Trilogy with AWK, CRYPT & Korn Shell		119	99
MKS VI Editor by MKS		75	65

novell development products

Btrieve/ISAM Mgr with No Royalties		245	184
Xtrieve Query Utility		245	184
Report Option for Xtrieve		145	99
Btrieve/N for Networks		595	454
Xtrieve/N		595	454
Report Option/N for Xtrieve/N		345	269
XQL		795	599

peter norton products

Advanced Norton Utilities		150	89
Norton Commander by Peter Norton		75	55
Norton Guides Specify Language		100	65
Norton Utilities by Peter Norton		100	59

phoenix products

Pasm86 Macro Assembler version 2.0		195	108
Pdisk Hard Disk & Backup Utility		145	99
Plantasy Pac Phoenix Combo		995	595
Pfinish Execution Profiler		395	209
Pfix86plus Symbolic Debugger		395	209
PforCe Specify C Compiler		395	209
PforCe+ + Specify C Compiler and C++		395	209
Plink86plus Overlay Linker		495	275
Pmaker Make Utility		125	78
Pmate Macro Text Editor		195	108
Pre-C Lint Utility		295	154

pmi products

EmsStorage	New	49	45
Graphix	New	149	119
Macro2		89	79
ModBase		89	79
Repertoire		89	74
Repertoire/Btrieve Toolkit	New	149	119

polytron products

PolyMake UNIX-like Make Facility		149	129
PolyShell	Special	149	105
PolyXREF Complete Cross Reference Utility		219	185
PVCS Corporate Version Control System		395	329
PVCS Personal		149	129

pop-up reference guides

Peabody by Copia Intl, Specify Language	New	100	89
Resident Expert Specify lang by Santa Rita	CALL	CALL	CALL

program mgmt utilities

Interactive EASYFLOW by Haventree		150	125
Quilt Computing Combo QMake & SRMS		250	199
TLIB Version Control System by Burton	Sale	100	79

sco products

XENIX System V for PS/2 by SCO	New	CALL	CALL
XENIX System V 286 by SCO		1295	979
Development System		595	479
Operating System		595	479
Text Processing Package		195	144
XENIX System V 386 by SCO		1495	1145
Development System		695	589
Operating System		695	589
Text Processing Package		195	144

software garden products

Dan Bricklin's Demo II by Software Garden		195	179
Dan Bricklin's Demo Pgm by Software Garden		75	57
Dan Bricklin's Demo Tutorial by Software Garden		50	45

TurboGeometry Library by Disk Software

List \$100 Ours \$89

TurboGeometry contains over 150 routines that perform geometric calculations. Topics include: intersection of lines, arcs, circles and planes; finding coefficients of line equations, planes and circles; distance between points, lines, circles, arcs, and planes; decomposition of polygons; 2D/3D transformations; 2D/3D curve generation, vector computations; convex hull computations; and much, much more.

texas instruments

Arbort Decision Tree Software		595	519
PC Scheme Lisp		95	77
Personal Consultant Easy		495	435
Personal Consultant Images		495	435
Personal Consultant Online		995	869
Personal Consultant Plus		2950	2589
Personal Consultant Runtime		95	84

text editors

Brief & dBrief Combo from Solution Systems		275	CALL
Brief		195	CALL
dBrief Customizes Brief for dBASE III		95	CALL
Epsilon Emacs-like editor by Lagan		195	147
Vedit Plus by CompuView		185	128

turbo pascal utilities

Azatar DOS Toolkit by Azatar		95	85
Tmark by Tangent Designs		80	69
TurboGeometry by Disk Software	New	100	89
TurboHALO from IMSI	Sale	95	69

turbopower products

TDEBUG 4.0		45	39
Turbo Analyst 4.0	New Version	75	59
Turbo Professional 4.0	New Version	99	79
TurboPower Utilities		95	78

other products

ACTOR by The Whitewater Group	New	495	419
ApBasic by CompTech Software	New	99	79
DMS Screen Master American Software Intl	New	200	185
DocusMotion by NWP-Intelligent Solutions	New	160	139
Heap Expander by The Tool Makers	New	60	55
Hi-Screen XL by Softway	New	149	119
MASTER*KEY by Sharpe Systems	New	80	69
Opt-Tech Sort by Opt-Tech Data Proc		149	99
Pascal-2 by Oregon Software	New	395	289
pcAnywhere by EKD Computer	New	99	89
SCREENIO for Realia by Norcom		400	379
Teamwork/PCSA by Cadre Technologies	New	995	929
TeleSwitch by EKD Computer	New	289	229
XenoCopy-PC by Xenosoft		80	69
XenoFont by Xenosoft	New	50	45

CALL for Additional Products

SWAINE'S FLAMES

I believe that the pace of change in programming methodology is increasing, and that those who learn new paradigms will keep up, while those who do not will be left behind.

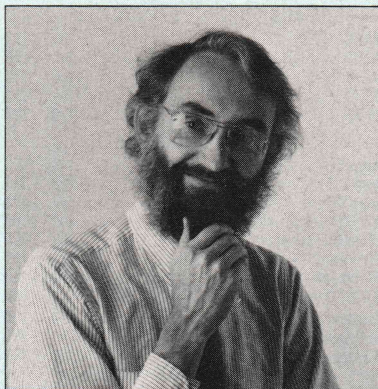
I am encouraged in this belief by a rereading of Robert Floyd's acceptance lecture on receiving the 1978 Turing Award. Floyd contended then that "continued advance in programming will require the continued invention, elaboration, and communication of new paradigms."

Floyd took the term *paradigm* from Thomas Kuhn's *The Structure of Scientific Revolutions* and used it to refer to general models of problem solving and the shared conventions and traditions of a discipline. Structured programming is a general paradigm, recursion a narrower one.

In his lecture, Floyd described how he invents new paradigms. Having solved a problem, he next resolves the problem from scratch, then looks for the general rule for solving problems of this sort, ultimately deriving a broad problem-solving paradigm. "Most of the classical algorithms to be found in texts on computer programming can be viewed as instances of broader paradigms," Floyd said. "Simpson's rule is an instance of extrapolation to the limit. Merge sorting is an instance of the divide-and-conquer paradigm. For every such classic algorithm, one can ask, 'How could I have invented this,' and recover what should be an equally classic paradigm."

And that's what you need to do if you want to advance the field and your place within it, according to Floyd: "I believe that the best chance we have to improve the general practice of programming is to attend to our paradigms."

I believe that attending to our paradigms is more imperative today than a decade ago, for two reasons.



First, there are simply more paradigms that we must understand today. Consider this list of vogue topics: logic programming, production systems, expert systems, black-board systems, functional programming, object-oriented programming, event-driven programming, neural nets, associative memory models, machine learning paradigms, MIMD, SIMD, and data-flow programming. Many of these topics overlap and some may be synonyms, but how many programmers can sort them out? Or predict which will be important two years from now?

Second and ultimately more far-reaching, I believe that a very deep paradigm, the Von Neumann model of sequential processing, is in the process of being supplanted by many parallel-processing paradigms. If this is true, it will be the most fundamental change in programming since the development of high-level languages, and it will radically affect the way we think about the process of writing software. With certain limited and constrained exceptions, all software is written within the Von Neumann paradigm. As programmers we scarcely know how to think in parallel terms. Our algorithms will not transfer. We will need a paradigmatic approach in order to find our way in a parallel world.

The parallel world is bigger, and hairier, than the sequential world.

True, there are grounds for skepticism about parallel processing. There is no architectural platform

for parallel programming outside certain specialized areas, such as numerical analysis and graphics. The kind of parallel processing I am talking about—multiple instruction, multiple data (MIMD) programming—is difficult, with significant unsolved problems.

True, there is no parallel equivalent of the IBM PC. But desktop multiprocessor architectures based on the transputer chip exist as commercial products today; they are just not yet cost-effective. The biggest cost factor is the price of the transputer, which is a function of demand and competition. This barrier could start falling within the year. It is not too early to imagine what you could do with a true parallel-processing system.

True, decomposition of a problem into MIMD parallelizable components is hard. That is precisely why a paradigmatic approach is necessary: finding the right paradigm can give you the solution to a broad class of problems. The divide-and-conquer paradigm, for example, is well adapted to MIMD parallelization, so if you can cast your problem in that form, you should be able to find a good parallel solution. And a good parallel solution is one that increases throughput radically.

I believe that the most successful programmers in the next decade will be those who carry in their toolkits, among their shiny metric and nonmetric algorithms, a rich set of paradigms.

Michael Swaine

Michael Swaine
editor-in-chief

All the Tools You Need For Motorola 680X0 From Whitesmiths

Whitesmiths, Ltd. now offers a complete set of 68K Cross Development Tools — specifically designed to work together — for the Motorola 68000 family of microprocessors. You get:

A C CROSS COMPILER

Whitesmiths' C Compilers offer the closest conformance currently available to the draft ANSI C Standard. We've added 68020 and 68881 support, and dramatically optimized code generation, so you can get the code quality you need today with the language you'll need tomorrow.

SUPPORT TOOLS

We have all the extras you need to develop embedded programs. Our powerful object utilities help you link multi-segment programs, build direct and sequential libraries, create load maps and interspersed listings, and talk to dozens of downloaders, emulators, and PROM programmers.

C SOURCE LEVEL DEBUGGING

We have the support you need to debug in terms of C functions, data types, and source lines. You debug what you write, not a lower level language.

A MICSIM SIMULATOR

You can debug your embedded programs right on your development host — our MICSIM Simulator needs no extra hardware. It's like debugging on your favorite emulator, but with no contention for dedicated resources, no download time, and with the symbolic breakpoint and trace control you've always dreamed of having.

AN XA8 CROSS ASSEMBLER

Our macro assembler is both fast and powerful, with support for 68020, 68881, and 68551.

A PASCAL COMPILER

You can program as much as you want in ISO Standard Pascal, or use the powerful extensions we've added to this production quality compiler. And you get complete integration with C and assembly language as well.

Working together, the 68K Cross Development Tools deliver both optimized performance and improved programmer productivity. Best of all, Whitesmiths offers everything you need at a very competitive price. We've been delivering and supporting high quality software development tools since 1978, and we're committed to continually enhancing our product line.

If you develop 68000 programs on a DEC VAX, an IBM PC, or a UNIX workstation, chances are we can save you time and money. For more technical details, call our toll-free number today. We also offer attractive packages for OEMs.



Whitesmiths, Ltd.

59 Power Road
Westford, MA 01886
617/692-7800

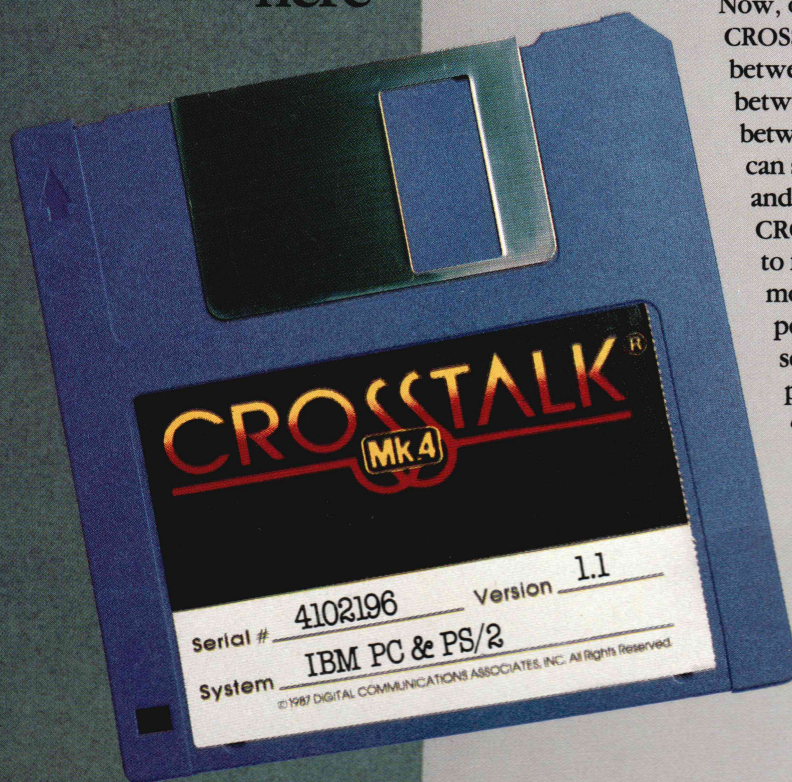
TOLL-FREE
NUMBER
1-800-225-1030

IBM
Spoken
Here

and
here

and
here

and
here



**Whatever dialect of IBM you need to speak,
CROSSTALK® Mk. 4 makes the connection.**

Now, one program does the job that used to require several. CROSSTALK® Mk. 4 allows high-speed direct communications between PCs and minicomputers, or (with an IRMA™ board) between your PC and an IBM Mainframe, or (with Smart Alec™) between your PC and IBM System 3x's. If you like, CROSSTALK can support all of these sessions (and others) simultaneously, and display each session in its own window.

CROSSTALK Mk. 4 emulates all the terminals you're likely to find useful. That includes IBM 3101 (page and character modes), IBM 525x, IBM 529x, IBM 327x, as well as many popular async terminals like the DEC VT100 and VT220 series. CROSSTALK Mk. 4 includes the powerful CASL™ programming language, which allows you to automate communications applications quickly and easily.

So if you're used to thinking of CROSSTALK just to use with a modem, you're missing some important connections. Ask your dealer for details, or write:

dca® Digital Communications Associates, Inc.
1000 Holcomb Woods Parkway / Roswell, Georgia 30076
1-800-241-6393

CROSSTALK®
COMMUNICATIONS

CROSSTALK and DCA are registered trademarks of Digital Communications Associates, Inc. IRMA, Smart Alec and CASL are trademarks of Digital Communications Associates, Inc. IBM is a registered trademark of International Business Machines Corp. DEC is a registered trademark of Digital Equipment Corp.

CIRCLE NO. 201 ON READER SERVICE CARD